REPORT DOCUMENTATION PAGE

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE October 8, 1996	3. REPORT TYPE AND DATES COVERED Quarterly Technical Report

4. TITLE AND SUBTITLE

Scientific and Technical Report
Intelligent Metacomputing Testbed

5. FUNDING NUMBERS

F19 628-96-C-0020

6. AUTHOR(S)

Dan Bender   Reagan Moore
Richard Marciano
Jay Boisseau

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

San Diego Supercomputer Center
PO Box 85608
San Diego CA 92186-5608

8. PERFORMING ORGANIZATION REPORT NUMBER

N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

ARPA/ITO
Att: Gary Koob
3701 N. Fairfax Drive
Arlington VA 22203-1714

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

N/A

11. SUPPLEMENTARY NOTES

DISTRIBUTION STATEMENT A
Approved for public release
Distribution Unlimited

19961016 048

12. DISTRIBUTION/AVAILABILITY STATEMENT

ARPA Agent
ARPA/ITO
DCAA San Diego North County Office

DTIC QUALITY INSPECTED 4

DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

The Distributed Object Computation Testbed (DOCT) has two principal goals: the demonstration of an object computation environment that supports distributed processing of large archived data sets and the demonstration of support for electronic submission and processing of complex documents and patent applications for the U.S. Patent and Trademark Office (USPTO). The infrastructure that is being integrated to create this testbed includes archival storage systems, databases, an object computation system, document management systems, and intelligent agents that support the patent application workflow. The resulting technologies should also apply to the information needs of other agencies, such as the National Science Foundation, the National Institutes of Health, the Nuclear Regulatory Commission, the Environmental Protection Agency (EPA), the Department of Energy, and the Department of Defense.

14. SUBJECT TERMS

Data Handling, Data Mining, Distributed Computing

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	

Intelligent Metacomputing Testbed
(Distributed Object Computational Testbed (DOCT))

San Diego Supercomputer Center
Reagan Moore, Principal Investigator

QUARTERLY SCIENTIFIC & TECHNICAL REPORT

July 1996 - September 1996

Sponsored By:
Advanced Research Projects Agency/ITO

ARPA Order No. D570
Issued by ESC/ENS under contract F19628-96-C-0020

Table of Contents

1. Task Objectives

The Distributed Object Computation Testbed (DOCT) has two principal goals: the demonstration of an object computation environment that supports distributed processing of large archived data sets, and the demonstration of support for electronic submission and processing of complex documents and patent applications for the U.S. Patent and Trademark Office (USPTO). The infrastructure that is being integrated to create this testbed includes archival storage systems, databases, an object computation system, document management systems, and intelligent agents that support the patent application workflow. The resulting technologies should also apply to the information needs of other agencies, such as the National Science Foundation, the National Institutes of Health, the Nuclear Regulatory Commission, the Environmental Protection Agency (EPA), the Department of Energy, and the Department of Defense.

The DOCT project consists of a collaboration of eight research organizations, led by the San Diego Supercomputer Center. The participating organizations include:

- California Institute of Technology (Caltech)
- National Center for Supercomputing Applications (NCSA)
- Old Dominion University (ODU)
- Open Text Corporation (Open Text)
- Science Applications International Corporation (SAIC)
- University of California at San Diego (UCSD)
- University of Virginia (UVa)

The DOCT project will investigate the creation of an innovative, distributed, national-scale, persistent document handling that will be capable of manipulating, searching, and managing terabytes of compound, complex, mixed-mode documents and data sets. This system will provide the underlying infrastructure for supporting the document management services needed for electronic commerce applications. The DOCT hardware/network systems will consist of high performance computing, communication, and storage devices distributed nationwide at university and government sites from coast to coast.

The DOCT software infrastructure will be created by integration of the following software layers and associated functions:

- Intelligent agents.
- Document management system.
- Applications.
- Persistent object computation system.
- Application-level scheduling system.
- Communication system.
- Data handling system.
- Object-relational database management system.
- Archival storage system.

In total, there are 8 Primary Task Groupings, which are further subdivided into approximately 50 tasks, or work elements. The 8 Primary Task Groupings are:

A) Develop and document data models

B) Establish metacomputing testbed

C) Develop archival storage and retrieval systems

D) Integrate object computation and data handling systems

E) Define assured service availability & fault tolerance/security requirements

F) Develop resource management & load sharing requirements

G) Develop intelligent software agents

H) Implement planning & management tools and procedures

The following table represents a list of tasks completed in FY96.Q4. For detailed task objective decriptions, see Appendix A.

Task - Task Description	Due
B2-1 - LiveLink Search Initial Installation and demo	9/30/96
B3-1 - Legion Initial Installation and demo	9/30/96
B4-1 - Analysis of alternate DMS (White Paper)	9/30/96
C2-1 - Tech report: File I/O interface spec. revision	7/30/96
C2-2 - Integrate Postgres with UniTree	8/29/96
G1-1 - Determine Agent Framework Requirements in the DOCT	9/30/96
H1-1 - Initial RDPS	9/30/96

Table 1-1: Tasks Completed in FY96.Q4

2. Technical Problems

The technical problems faced during completion of the tasks for FY96.Q4 (listed in Table 1-1) are listed in Appendix B.

3. General Methodology

3.1 Technical Methodology

The DOCT project will investigate the creation of an innovative, distributed, national-scale, persistent document handling sytsem that will be capable of manipulating, searching, and managing terabytes of compound, complex, mixed-mode documents and data sets. This system will provide the underlying infrastructure for supporting the document management services needed for electronic commerce applications. The DOCT hardware/network systems will consist of high performance computing, communication, and storage devices distributed nationwide at university and government sites from coast to coast (see figure 3-1). The DOCT software infrastructure will be created by integration of the following software layers and associated functions using a philosophy of rapid prototyping:

- Intelligent agents. Processing a patent application requires specifying many diverse actions that will be taken at each stage of the process. DOCT will provide an easy to manage mechanism for this requirement by developing intelligent agents that are linked together within a common framework. The agents will operate within the Legion environment, and take full advantage of the object structure provided by Legion.

- Document management system. Patent applications are typically compound documents with individual sections that require separate processing steps. DOCT will integrate a document management system into the Legion environment to manage the flow and control the processing of complex documents and their components

- Applications. Multiple applications will be supported for additional tasks such as text search and pattern recognition. These applications will function within the Legion environment to enable their execution on any of the compute platforms within the DOCT testbed.

- Persistent object computation system. In a distributed environment, object persistence is very difficult to maintain. Legion, which supports processing of persistent objects between distributed computation platforms, will be used to control all data objects and track all modifications. Data sets modified on one compute platform will be available in the modified form for subsequent use on any of the other compute platforms within the testbed.

- Application-level scheduling system. Coordination of resource usage within a distributed environment requires the development of a distributed scheduling mechanism. The AppLeS scheduling system, which determines the best platform on which to execute an application given the current system workload and configuration, will be integrated with Legion. Each application will independently decide where it can be executed the fastest, based on information provided by AppLeS. Legion will then execute the application at the requested platform, and support all associated data movement.

- Communication system. The DOCT testbed is linked by heterogeneous networks, which run at different bandwidths and interconnect subsets of the compute platforms. Nexus, which supports transparent communication over heterogeneous networks, will be examined for integration with the Legion system. This will automate the transmission of data over the fastest available link, without requiring intervention by a user of DOCT.

- Data handling system. The data sources that will be linked by DOCT are distributed nationally across multiple types of storage systems. MDAS, which supports application access to arbitrary remote data sources, including databases, archives, and web servers, will be integrated with Legion. This will allow the persistent data objects maintained by Legion to be stored at any of the storage systems within the DOCT testbed.

- Object-relational database management system. Many of the data sets stored in DOCT can be identified mainly by their attributes which will be stored as metadata in a database. An ORDBMS supports query based identification of the data set of interest. Within the DOCT testbed, ORDBMS systems will be integrated with archival storage, to allow arbitrarily large collections of data to be accessed through a database interface.

- Archival storage system. When managing large collections of data, most data sets can be stored on tertiary tape devices at substantially lower cost. The High Performance Storage System (HPSS), which supports storage of data on heterogeneous peripheral disk and tape devices, migrates data between tape and disk to keep the most recently accessed data sets on the fastest storage devices. This allows fast response to most queries, without having to keep the entire data set on rotating disk.

3.1.1 Major Development Efforts

The major development efforts for creating the DOCT architecture are:

- Intelligent agents. Managing the work flow associated with processing a patent application requires specifying the actions that will be taken at each stage of the process, creating an intelligent agent to implement that action, and then linking the agents together within a common framework. The agents will be computation objects within the Legion environment, and access Legion data objects that are stored in the archive through a database interface.

- Distributed applications. To take advantage of this environment, applications will need to store all of their data objects through the Legion system. This will be accomplished by either wrapping the application with software layers that provide the interface to Legion, or by modifying the application code to directly interface to Legion data objects.

- Persistent object support for computation. Legion maintains a table with unique identifying tags for each data object (LOID) and its location in a data vault within the distributed environment. Data objects can be moved between data vaults and manipulated, without having to worry about their actual physical location. Objects may have copies in multiple vaults. DOCT will integrate both database and archival storage systems as separate vault types within Legion.

- Application Level Scheduling. AppLeS is a distributed scheduling system that is being developed by Berman and Wolski (UCSD), with partial funding from the DOCT project. An application identifies those resources it needs for execution, queries the available resources and networks for their expected load and processing times, and chooses that set of resources which will provide the minimum turn around time. AppLeS analyzes the load on the network to determine whether there is enough available bandwidth to move the data needed by the application. AppLeS will execute the application on the platform where the data is located if the available network bandwidth is to small.

- Transparent access across heterogeneous communication systems. Nexus (Kesselman, Caltech and Foster, ANL) is a system that optimizes data transfer when multiple types of networks are available. A collaboration will be sought with this group to include Nexus as a layer in the Legion system. A particular computer might be accessed over ATM, HiPPI, FDDI, or Ethernet networks. Nexus maintains tables of all the possible network access mechanisms for each computer in the distributed environment.

- Access of data sets by queries on metadata attributes. When an application is analyzing thousands of files, data management is only feasible using relational database technology. In order to support rapid data movement, large data sets will be transferred using third-party I/O directly from the archive to the parallel application using the MPI-IO standard. DOCT will integrate archival storage systems into object-relational database technology to support queries against very large data collections. The archive holds the data objects, while the database manages the access of the data.

- Storage of all data sets as persistent objects within the distributed archive. Legion maintains the class and object identifiers. The archival storage allows storage of data on either disk or tape.

3.1.2 Distributed Environment

The distributed environment that comprises the DOCT testbed is shown in Figure 2-1. This shows each of the participating sites and the network connections that will be used to link the sites together. The participating sites are SDSC, Caltech, NCSA, a Washington Area Metacomputing

4

site at the SAIC office in Arlington (WAM), University of Virginia, and ODU. Replicated archival storage systems will be established at SDSC and Caltech. Database systems will be installed at SDSC, NCSA, and the WAM. Compute servers will be provided at SDSC, NCSA, and the WAM. Two transcontinental networks will be used to link the sites. SDSC, NCSA, Caltech, UVa, and ODU will be linked by the very High-speed Backbone Network Service (vBNS) which is being upgraded to support communication at 622 Mbps. SDSC will also be linked to the ATDnet in Washington DC across AAInet. This network is being upgraded to support communication at 155 Mbps. SDSC is installing a local link at 155 Mbps to the Naval Command and Control Ocean Surveillance Center which is a node on AAInet. To provide completely redundant communication paths, a connection between the vBNS and the ATDnet is needed in Washington DC.
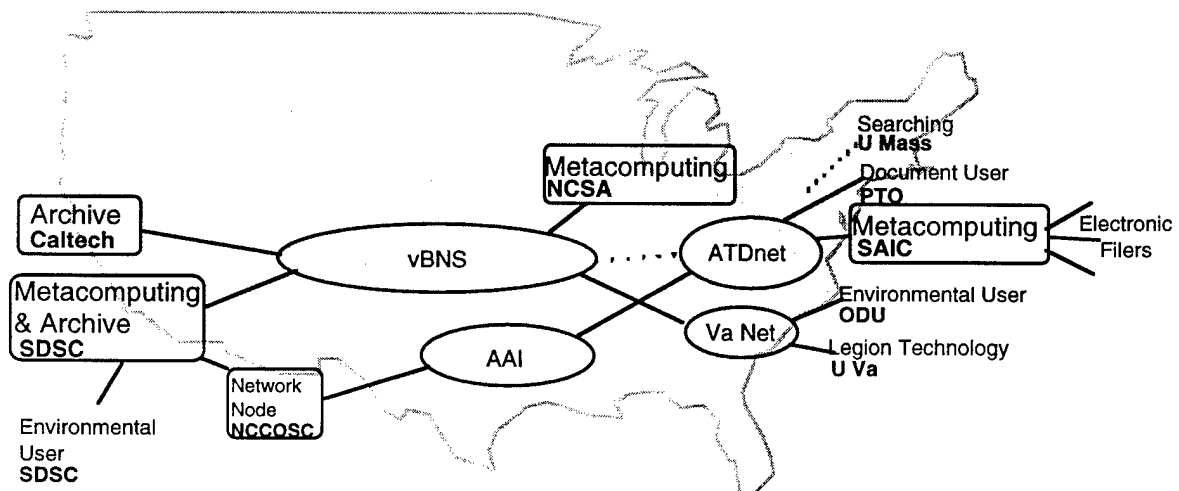
Figure 3-1: Distributed Object Computation Testbed network interconnects

3.1.3 Approaches to Testbed Integration

DOCT will explore two approaches to integrating distributed computation and data. The preferred approach is characterized by tight coupling between the software sub-systems and the object-oriented environment. In this model, all applications obtain data from objects using the Legion Object Identifier. This ensures that only one sub-system (Legion) is managing the data object and that the system will never lose track of the object or related information (maintain object persistence). Tight coupling usually requires that the data reference methods of a sub-system be modified to use an object oriented external data store mechanism.

The less preferred method, loose coupling, is easier to achieve but less capable. In this approach, sub-systems are used as originally written and modify persistent data directly according to their original design (e.g. use of a file or database for persistent storage). An application might change an object (e.g. document) or metadata related to that object without the change being tracked by the Legion object management system. Subsequent references to the document (object) or its metadata by other than the original sub-system can result in the wrong version of the data being returned. To provide the integrity required of a Document Management System we will primarily use a tightly coupled approach.

We will create a data handling environment that supports compound, mixed mode, complex scientific data sets, which are then linked with the Legion persistent object computation system.

5

Since both systems are being designed as application level infrastructure, the environment will be portable across all of the DOCT platforms. The result will be a system that coordinates both computational object distribution and data access across the metacomputer, while each platform retains control of its individual workload and local file system. The tightly coupled version of the object computation architecture is shown in Figure 3-2. All applications (the DMS, search tools) access data objects through Legion. This implementation will have the advantage of being able to track any changes made to all data objects that are stored in either the database or archive.
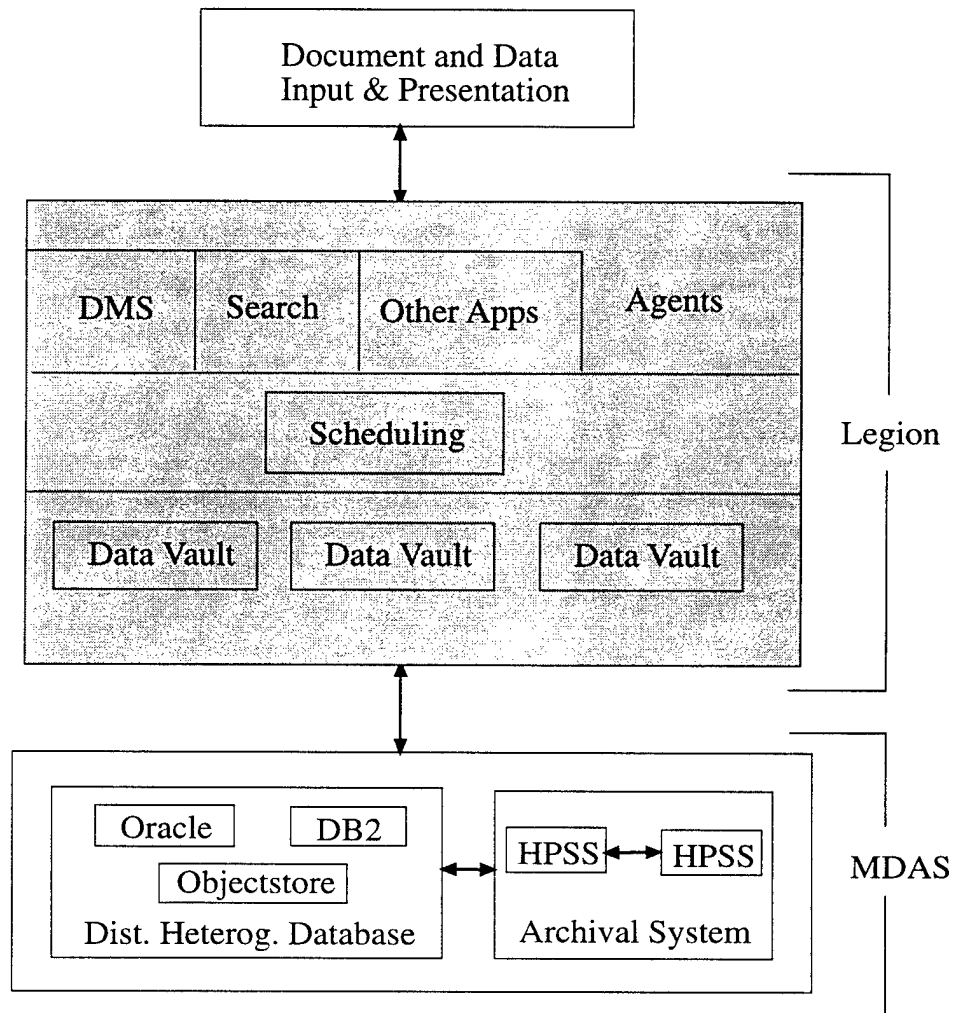
Figure 3-2: DOCT Tightly Coupled Architecture

In contrast, a more loosely coupled architecture is shown in Figure 3-3. In this case, the applications such as the text search engine directly access data without going through the Legion environment. The implication is that an intelligent agent will need to guarantee that every time the approved patent database is updated, the text search engine will be notified to revise its search index. Similarly, any other application which access data without using Legion LOIDs will have to coordinate their local data sets with the Legion persistent data sets.
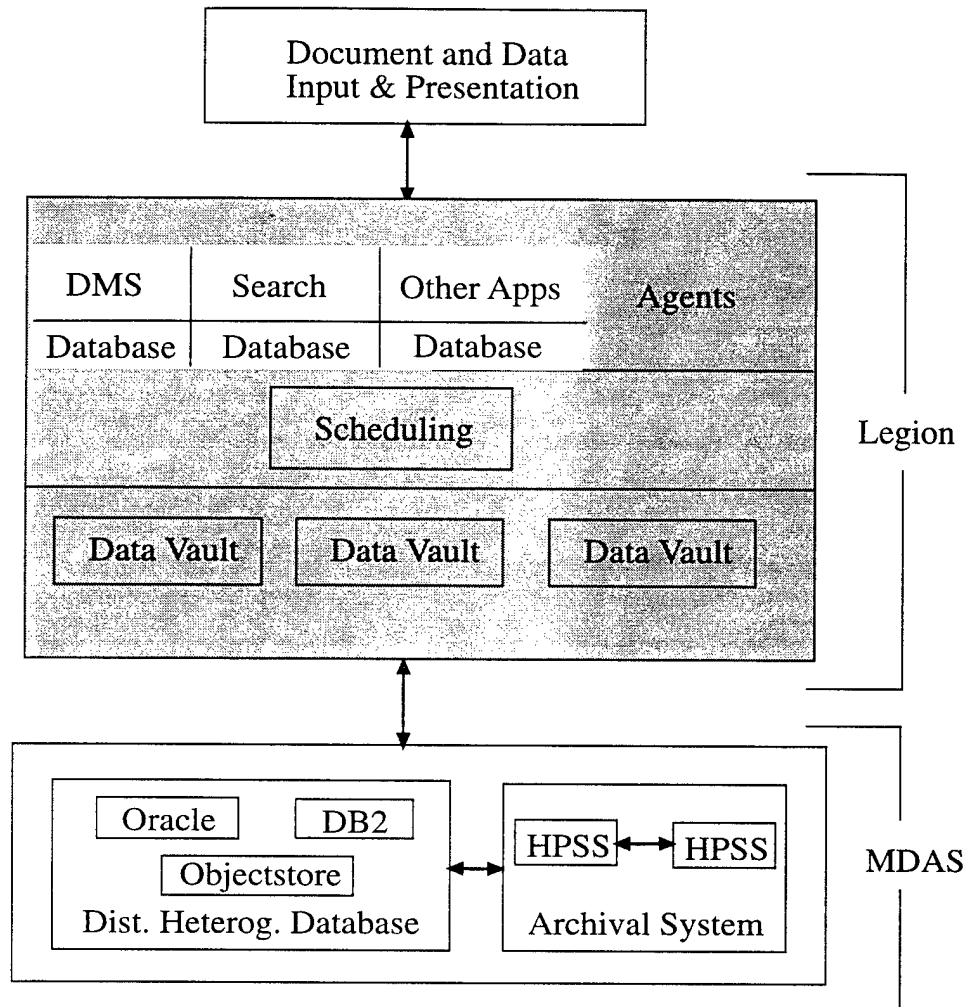
Figure 3-3: DOCT Loosely Coupled Architecture

3.2 Management Methodology

To facilitate the management of the DOCT project, the DOCT project has been divided into 8 primary work breakdown structures (task groupings) and the DOCT participants have been divided into 11 key technical areas.

The 8 primary task grouping include:

A. Develop document and data models
B. Establish metacomputing testbed
C. Develop archival storage and retrieval systems
D. Integrate object computation and data handling systems
E. Define assured service availability, fault tolerance/security requirements
F. Develop resource management and load sharing requirements
G. Develop intelligent software agents
H. Implement planning and management tools and procedures

7

The 11 key Working Groups are as follows:

WG1 - Steering Committee Formation
WG2 - Architecture
WG3 - Electronic Filing & Commerce
WG4 - Distributed Resource Management (Legion)
WG5 - Security
WG7 - Data Handling
WG8 - Intelligent Agents
WG9 - Environmental Data
WG10 - Text & Image Searching
WG11 - Videoconferencing

The DOCT management approach can be summarized in terms of three primary activities:
1) Report generation,
2) Technical and coordination meetings, and
3) Use of software management tools

Each management approach is described below.

3.2.1 Report generation

DOCT management reports include the following items:
- Monthly Progress report which summarizes significant activities in each of the 8 major task areas of the DOCT project
- Quarterly DARPA Progress report which is provided in the format specified by the DOCT Work Project Notification CDRL 0704-0188
- Research & Development Plan and Schedule (RDPS) which is a detailed plan, description, and schedule for all activities to be performed under this contract . The purpose of this document is three-fold: 1) to provide an integrated, detailed description of all the individual project tasks and deliverables for all DOCT participants and sponsors, 2) to provide a planning and scheduling tool to help in monitoring and evaluating the progress of DOCT, and, 3) to provide a baseline plan to control changes, i.e.. revised DOCT R&D priorities based on sponsor directives and ongoing R&D results and findings
- Concept of Operations (ConOps) document describes the DOCT test bed in terms of its features and capabilities, its relevance to HPCC in the coming decade, and opportunities for other federal agencies and DARPA projects to participate and/or collaborate
- The Final Summary Report is a combined technical and management report which will: a) summarize the technical progress made under this project, b) detail DOCT technologies ready for productization (commercialization), and c) detail DOCT technologies requiring additional R&D.

3.2.2 Technical and coordination meetings

The DOCT technical and coordination meetings include the following scheduled and ad-hoc gatherings and teleconferences:
- Weekly DOCT technical teleconferences: These are weekly meetings which address both technical and coordination issues regarding pending deliverables. The teleconference

participants include cognizant parties such as: lead task person, a representative from each DOCT subcontractor, and a representive from each relevant Working Group.

- Bi-weekly USPTO status teleconference: These meetings address technical status and management coordination issues particularly relevant to the USPTO tasks and deliverables.
- Quarterly status meeting: These meetings take place approximately 3 months apart to assess and monitor the status of the project. Participants include DOCT sponsors, subcontractors, steering committee members, and other interested parties.
- Steering committee meetings: The steering committee consists of other Federal Agencies and DARPA contractors that have expressed interest in participating in the Distributed Object Computation Testbed. These meetings are held on a quarterly basis.
- Ad-hoc meetings: These meetings are held as necessary, in person or by teleconference to address pressing issues relevant to the progress of the DOCT project.

3.2.3 Use of software management tools

The primary software management tools used by the DOCT project include the following:

- VTC tools: These Unix based software tools (VIC, VAT, WB) provide the capability for the DOCT participants to video teleconference and share technical documents of interest.
- Project resource and scheduling tools: The Microsoft Project software package is used to provide resource loading, scheduling, and task dependence reports for the DOCT project.
- Document management: Document management is provided by the Open Text Livelink document handling system, which control document access and workflow over the internet.

4. Technical Results

4.1 Testbed implementation status

The DOCT testbed consists of hardware resources located at 8 sites coupled by software and networks spanning the country (see Figure 3-1). The testbed implementation is largely complete, with the remaining resources to become available as listed in the Table 4-1 below.

HARDWARE

Site	CPUs	Comp.	Data Storage	Comp.	Network	Comp.
SDSC	SP-2 (17)	X	435GB DASD	X	vBNS	X
	Paragon (400)	X	NSL UniTree	X	OC3 (NCCOSC)	1-Oct
	C90	X	HPSS	X		
	T3D (128)	X	9TB Tape storage	X		
	T3E (256)	Jan-97	Raid upgrade	1997		
	Sparc 5	X				
SAIC	CS6400	X	50GB	X	ATDnet	15-Oct
	OCR Optical Power Server	X	300GB DASD	15-Oct		
	Large Screen Display	X				
NCSA	Challenge Array	X	200GB DASD	15-Oct	vBNS	X
	Exemplar	X				

9

Caltech	RS/6000	X	8TB Tape storage	1-Nov	vBNS	15-Dec
USPTO	Sparc 5 or HP	22-Oct			ATDnet	1-Nov
ODU	SGI workstation	X			vBNS	1-Nov
UVa	SP/2 (14)	X			vBNS	1-Jan
	PC	Jun-97				

SOFTWARE

Site	Databases & Data	Comp.	Legion	Comp.	LiveLink Search	Comp.	LiveLink DMS	Comp.
SDSC	Postgres95	X	v2.9	X	v.6	15-Oct	4.0.3	15-Oct
	Illustra	X						
	DB-2	X						
	2GB Messenger data set	?						
SAIC	ObjectStore	X	v2.9	X	v.6	X		
	2GB Messenger data set	X						
	40Gb Messenger image data	1-Jan						
NCSA	Oracle	X	v2.9	X	v.6	X		
	Informix	X						
Caltech	HPSS	X	v2.9	X				
USPTO			v2.9	1-Nov				
ODU			v2.9	1-Nov				
UVa			v2.9	X				

Table 4-1: Testbed implementation schedule

4.2 Scheduled Deliverables

4.2.1 Reports

The following reports were completed during FY96.Q4:

MP ID #	Task ID & Description	Scheduled Start Date	Scheduled Delivery Date	Lead Person
75	C2-1 - Tech report: File I/O interface specification revision	7/1/96	7/30/96	Baru
146	G1-1 – Determine Agent Framework Requirements in the DOCT Environment	7/1/96	9/30/96	Wade
50	B4-1 - Analysis of alternate DMS (White Paper)	7/1/96	9/30/96	Mosier
162	H1-1 - Initial RDPS	7/1/96	9/30/96	Bender

Table 4-2: Reports completed in 3Q96

For detailed task descriptions and technical issues, refer to appendices A and B, respectively.

4.2.2 Demonstrations

The individual task demonstrations that were performed in FY96.Q4 are listed below:

MP ID #	Task ID & Description	Scheduled Start Date	Scheduled Delivery Date	Lead Person
76	C2-2 - Integrate Postgres with UniTree	7/1/96	8/29/96	Baru
44	B2-1 - Livelink Search Initial Installation and demo	7/1/96	9/30/96	Kesselman
47	B3-1 - Legion Initial Installation and demo	7/1/96	9/30/96	Grimshaw

Table 4-3: Demonstrations completed in 3Q96

For detailed task descriptions and technical issues, refer to appendices A and B, respectively.

5. Important Findings and Conclusions

The DOCT project has an 18-month schedule during which all deliverables will be produced. The deliverables of this project are technical reports and demonstrations. The first of these deliverables are just now being completed. As more deliverables are produced, we will list important findings and conclusions in subsequent quarterly reports.

6. Significant Hardware Development

This section is not applicable to the DOCT project since hardware development is not part of the contracted scope of work.

7. Special Comments

We have written a Research and Development Plan & Schedule (RDPS) document for this project. The RDPS is a detailed, comprehensive overview of all DOCT tasks, including all deliverables (reports and demonstrations). This document is the authoritative source for our project goals, tasks, deliverables, schedule, and expected work loads.

The RDPS is available via the World Wide Web at the URL listed below:

http://www.sdsc.edu/DOCT/RDPS/rdps-doct.html

The RDPS can be viewed in HTML form. The Microsoft Word v6 version and PostScript version can be downloaded to the reader's local workstation/personal computer.

We have included two appendices as part of the DARPA quarterly report:
- Appendix A: Task Descriptions
- Appendix B: Task Issues

These appendices include information pertinent to the the entire project and provide a baseline reference for the tasks and issues discussed in this (and future) quarterly report.

8. Implications for Further Research

The DOCT project will establish a testbed infrastructure designed specifically to further research into the following areas of distributed computing:
- Intelligent agents
- Document management systems
- Applications
- Persistent object computation systems
- Application-level scheduling systems
- Communication systems
- Data handling systems
- Object-relational database management systems
- Archival storage systems

The DOCT management team is writing a Concept of Operations (ConOps) document which describes the DOCT testbed in terms of its features and capabilities, its relevance to HPCC in the coming decade, and opportunities for other federal agencies and DARPA projects to participate and/or collaborate. This document specifies both ongoing research efforts and future R&D efforts relevant to multiple Federal Agency missions. This document is scheduled for completion by November 1996. Excerpts from the completed document will included in the next DARPA Quarterly Report.

APPENDIX A: Task Descriptions and Objectives

In total, there are 8 Task Groups, which are further subdivided into approximately 50 tasks, or work elements.

A) Develop and document data models

B) Establish metacomputing testbed

C) Develop archival storage and retrieval systems

D) Integrate object computation and data handling systems

E) Define assured service availability & fault tolerance/security requirements

F) Develop resource management & load sharing requirements

G) Develop intelligent software agents

H) Implement planning & management tools and procedures

Each task and subtask description and objective is provided below.

A. Develop and document data models

- **Task A1 - Analyze Open Systems and Industry Approaches to Represent Complex Work Units:**

    This task will analyze open systems and industry approaches to represent complex work units in electronic format and develop a Technical Report, which is intended to serve as a roadmap for future development of electronic filing and electronic commerce systems applicable across a wide class of federal agencies.

- **Task A2 - Evaluate Hardcopy Scanning, Conversion, and Translation Methods:**

    This task will evaluate various advanced approaches for scanning, converting, and translating hardcopy, paper-based, scientific and technical documents using optical character recognition (OCR) techniques. Specifically, the capabilities of the SAIC Optical Power Server (OPS) will be enhanced to recognize scientific and technical expressions, particularly mathematical and chemical expressions which can be represented and tagged in ASCII. OPS will also be enhanced to perform improved OCR on technical information contained within tables. The major products of this task will be a Feasibility Report and two (2) demonstrations of OPS enhanced to handle documents containing mathematical and chemical expressions, as well as tables. For demonstration purposes, sample documents to be scanned and converted will include representative paper-based patent application documents; as well as published patents (especially images for granted patents prior to 1971.)

- **Task A3 - Explore SGML Autotagging Technology:**

    This task will explore the use of SGML autotagging technology to convert the scanned output of an OCR device to SGML-tagged format. Additionally, the task will consider conversion and autotagging of word processing and display formats to SGML format. Lastly, the task will gain practical experience and develop metrics for conversion and autotagging of documents in standard formats in support of the *Draft Implementation Guide for Electronic Filing* and in support of the published backfile conversion problem.

- **Task A4 - Develop and Evaluate a STEP-Based Approach to Representation of Intellectual Property:**

     This task will analyze and assess the potential applicability of the ISO 10303 STEP standard (STandard for the Exchange of Product Model Data) as a method for representing and interchanging information about intellectual property described within a patent application. This task will develop a Technical Report assessing STEP's potential for patent application data delivery. As feasible, the task will also develop and/or adapt existing STEP Application Protocols (APs) to support patent applications in an intelligent electronic format. A demonstration of STEP technology may also be possible.

- **Task A5 - Develop a Validation Mechanism to Support Electronic Filing:**

     This task will prototype and demonstrate the Validation Mechanism concept as discussed in the *Draft Implementation Guide for Electronic Filing* within the DOCT environment.

Tasks A6-A8 will explore the potential application of Virtual Reality Modeling Language (VRML) technology for complex intellectual property documents as well as alternative environmental data sets. The VRML tasks will address validation of VRML content as well as long-term implications associated with retrieval and display of VRML archives.

- **Task A6 - Develop VRML Submission Support:**

     This task involves examination of the process for submitting VRML content as part of an archived document set, such as for a patent or trademark submission, or inclusion in an archived environmental data set. This submission process requires development of software to retrieve, bundle, and validate all VRML files in a submission, together with their attendant images, movies, sounds, and animation scripts.

- **Task A7 - Examine VRML Retrieval:**

     This task involves examination of the process for retrieving previously-archived VRML content. This retrieval process requires consideration of software to do 3-D content-based retrieval, including attempts to extract semantic information from VRML content. This work will analyze the issues involved and make recommendations in the form of a White Paper.

- **Task A8 - Examine VRML Display:**

     This task involves examination of the process of rendering and displaying previously-archived VRML content. This process requires consideration of the variability in display techniques for VRML, and how that variability may affect the interpretation of VRML content. This is potentially important consideration in evaluating and interpreting intellectual property containing VRML files. The rendering and display process also requires the appropriate hardware and software which need to be available at the time of content retrieval. In the case of a patent or trademark database, VRML display capability needs to exist for, at least, the duration of the patent (or trademark). For all intents and purposes, this time frame is effectively on the order of the lifetime of our social institutions. This issue is also of vital interest and concern to the National Archives and Records Administration (NARA). This task work will analyze the major issues involved in displaying VRML content over the lifetime of the archive and make recommendations in the form of a White Paper.

Tasks A9-A11 will demonstrate the ability of DOCT to handle and integrate environmental data from the Chesapeake Bay and San Diego Bay. T

- **Task A9 - Model and Represent Environmental Data:.**

    This task deals with the construction and integration of environmental data into the infrastructure of DOCT and is an area of active ongoing research at SDSC, ODU, and NCSA. Exploratory research will be done on interchange, search and query, and visualization of environmental data sets of the Chesapeake Bay and San Diego Bay, including bathymetry data, numerical simulations, and hydrographic and chemical observations.

- **Task A10 - Integrate Environmental Database with Archive:**

    This task deals with the integration of environmental data with database and archival storage technologies. Exploratory research will be conducted on supporting storage, querying, searching, and use of metadata for environmental data.

- **Task A11 - Integrate Environmental Database with Legion:**

    This task deals with the integration of environmental data with the Legion environment. Exploratory research will be conducted on supporting persistent environmental objects that can be manipulated and retrieved in a distributed environment.

- **Task A12 - Legacy Data Migration and Data Load Module:**

    This subtask will analyze the basic systems engineering and logistics requirements for migration of the Messenger text database and the Classified Search and Image Retrieval (CSIR) image database to the DOCT environment. Additionally, the subtask will develop the basic approach and initial high-level, database load module to support Messenger and CSIR data within DOCT. Lastly, this subtask will address requirements for, and availability of, legacy trademarks data in DOCT.

- **Task A13 - Prototype the *Draft Implementation Guide for Electronic Filing* with Intellectual Property Community Participation:**

    This subtask will prototype and demonstrate major elements of the *Draft Implementation Guide for Electronic Filing* with subcontractor support from a firm (or team) with broad and recognized experience in the following areas:

    - electronic document authoring systems
    - open systems standards for intellectual property document interchange
    - open systems and industry approaches to represent complex work units in electronic format
    - application of SGML, STEP, VRML, and intelligent graphics formats in documents
    - legal and regulatory aspects of electronic commerce and electronic filing including requirements for validation
    - the application of the Internet, and
    - hands-on experience with the patent and trademark application and prosecution process including amendment processing.

- **Task A14 - Document Classification using Self-Organizing Maps:**

    This demo task will illustrate the use of Kohonen's self-organizing feature maps for automatic classification and searching of patent documents.

15

B. Establish metacomputing testbed

- **Task B1 - Install Testbed Infrastructure:**

     This task involves the initial installation of Equipment, Software, Accounts, Initial Data, and Network infrastructure in support of DOCT. Primary equipment components (e.g., host computers) are already in place, but additional disk will be added. While the major DOCT software components are included in other tasks within this group (i.e. B2-B6, and B8), the software installation covered in this task includes miscellaneous and infrastructure support software. This task also includes the creation of user accounts for DOCT participants on all testbed computers and the loading of initial test data. Network infrastructure includes initial connectivity and upgrades to higher performance links.

- **Task B2 - Installation, Development, and Integration of Livelink Search:**

     This subtask provides for the initial installation of Open Text "Livelink Search" on metacomputing testbed platforms located at SAIC, SDSC and NCSA. Additionally, the subtask provides for the installation of upgrades to Open Text Livelink Search which enhance its text search, information retrieval, and viewing capability over the period of performance of DOCT.

- **Task B3 - Install and Update Legion System:**

     The Legion system is the software substrate that the DOCT project will employ to manage its distributed metacomputing testbed environment. Legion provides the functionality to create and manage objects in a distributed heterogeneous environment This includes managing the state of persistent objects, naming and binding to objects, and scheduling and instantiating objects on distributed resources.

     Since the development of Legion is ongoing, Legion will be installed on the metacomputing test bed in phases. The first task under B3 (B3-1) will be the installation of the initial version of Legion. This will be done as soon as the testbed infrastructure is in place. After the initial installation, enhanced versions will be installed on the testbed as additional, necessary functionality is developed within Legion (subtask B3-2). Some of this functionality will be developed as part of the DOCT project in collaboration with other members of the DOCT project team (e.g. the database-based Legion Vaults, see Task D1), while other functionality will be developed under separate funding.

- **Task B4 - Analysis of Alternate Document Management Systems:**

     This subtask will analyze, select, and install a fine-grained, enterprise-wide document management system (DMS) within the DOCT environment. A fine-grained document management system is defined to be a DMS which provides a capability to manage documents at the SGML element level of granularity. Accordingly, a fine-gained DMS approach has some significant future potential for streamlined amendment processing and version control in an electronic commerce environment.

- **Task B5 - Experiment with Existing Security Options:**

     This task involves examining and evaluating currently-available security technologies for possible deployment, demonstration and actual protection of the Metacomputing testbed.

- **Task B6 - Install Heterogeneous Database Environment:**

     A heterogeneous database environment is one in which there are multiple data sources (e.g. text data, image data, index data) and each data source employs a

16

different database system, or hardware platform, or data schema, or some combination of these variations to store its data. Individual applications that wish to access data from any one of these data sources should then be provided a means by which to do so. This is an important issue in the DOCT project since, for example, patent search applications may wish to search not just the USPTO data but various other data sources (digital libraries, other text databases and indexes, etc.) that are available in the distributed computing environment (e.g. Internet).

- **Task B7 - Load Patent and Trademark Data:**

  This task deals with the logistics of receiving legacy data from the USPTO, archiving this legacy data in the form received ("raw" data), transforming/indexing the data (e.g. by a text search engine for text data and by other pre-processing steps for image data), and the archiving of the transformed/indexed data.

- **Task B8 - Install Archival Systems:**

  This task involves installing the archival storage systems as part of the DOCT testbed. These include the High Performance Storage System (HPSS) from IBM, which will be installed on the IBM RS/6000 SP equipment at SDSC and Caltech, and the NSL UniTree which will be installed at SDSC. Archival storage systems provide storage capability to store terabytes to petabytes of data on-line. This provides a convenient facility for storing both raw and indexed patent and trademark data from the USPTO, as well as for storing data from other federal agencies which have requirements to store large amounts of data.

- **Task B9 - Develop Sample Test Agent:**

  This task will develop a simple, test software agent to help demonstrate that the metacomputing infrastructure is in place and functioning. A simple agent will be developed first for the initial infrastructure (Task B9-1). This agent may need to be refined as the functionality of the testbed evolves over time (Task B9-2).

C. Develop archival storage and retrieval systems

- **Task C1 - Develop data storage architecture:**

  Prior to loading any data in the archival system, it is necessary to decide upon a data storage architecture that provides the physical layout information regarding how the data will be stored within the archive, how individual data records will be formatted, what metadata, if any, will be stored with the archival data and, in the case of replicated archives, how the replication architecture will work. The issues addressed are general, though the specifics in each case will depend on the particular data being supplied by the USPTO. Some of the details may also vary based on the type of data, e.g. text vs. images.

- **Task C2 - Integrate Archival Storage Systems with DBMS:**

  The archival storage systems used in DOCT will be integrated with DBMSs to provide efficient access to metadata, to provide a uniform file I/O interface (implemented within the DBMS) to applications accessing the archival store, and to improve access speeds to archival data sets by caching data via the DBMS.

- **Task C3 - Experiment with Replication Alternatives:**

  To provide higher availability, the archival storage systems in DOCT support replication. Thus, if one archival site is not available for any reason, then another site can be used to service the request. When a data set is archived there are several options available for replicating this data for high availability. This task will study the various alternatives.

17

- **Task C4 - Use MDAS API's:**

  The Massive Data Analysis System (MDAS) is being developed as part of an independent DARPA-funded project at SDSC. We plan to leverage technology from this project to provide improved functionality in the DOCT testbed. Aspects of MDAS that are of interest to DOCT include the ability to discover and schedule resources in a distributed system, the ability to "connect to" these resources, and the ability to access remote data sets via standard interfaces. These capabilities are very useful and powerful, for example, for patent search and other patent applications that expect DOCT to offer an integrated metacomputing environment.

- **Task C5 - Integrate Extended MDAS Capabilities Into Legion:**

  An issue when storing and retrieving large data sets is the I/O bandwidth available to do this task. In addition, if the data sets are being read to/written from archival store (e.g. tape drives), the bandwidth may become even more restricted. Thus, it is essential to support parallel I/O capability between applications and archival store to speed up this task. Once again, we plan to leverage the work being done on the MDAS project to address the issue of providing parallel I/O capability to applications. With this capability, a patent/trademark application can read multiple documents or images in parallel. It can also read a single compound document or image in parallel as well.

  A corresponding extension is remote authentication of the application making the I/O request to the data storage system.

D. Integrate object computation and data handling systems

- **Task D1 - Legion Integration with Databases:**

  This task will analyze methods for integrating database technology into the Legion persistent object computation environment, implement the best method, and demonstrate the successful storage, retrieval, and modification of persistent data sets that are managed by a database.

- **Task D2 - Legion Integration with archival storage:**

  This task will analyze methods for integrating archival storage technology into the Legion persistent object computation environment, implement the best method, and demonstrate the successful storage, retrieval, and modification of persistent data sets that are managed by a hierarchical storage system.

- **Task D3 - Integrate Livelink Search with Legion:**

  This task will integrate the Open Text Livelink text search engine with the Legion persistent object computation environment. This will be done as a tightly coupled system in which the text search engine directly references relevant Legion objects when constructing the search index.

- **Task D4 - Integrate Document Management System with Legion:**

  A document management system (DMS) will be integrated with the Legion persistent object computation environment. The intent is to allow the DMS to manage the compound document work flow processing, while maintaining the persistent objects through the Legion data access interfaces. Thus all changes resulting from DMS operations need to be turned into new objects stored within the Legion data vaults.

- **Task D5 - Develop architecture analysis:**

  This task will analyze the tightly coupled architecture that is being developed and propose alternate systems that can be used to mitigate risk.

E. Define assured service availability & fault tolerance/security requirements

Tasks E-1, E-2 and E-3 are analysis tasks which will result in the publication of the Security Risk Analysis and Plan deliverable required under proposal Task 1D.

- **Task E1 -Analysis of Security Issues for Electronic Commerce Applications:**

  This task will examine the confidentiality, integrity, access control, authentication and availability of electronic commerce applications focused on elements of the USPTO *Electronic Filing Guide.*

- **Task E2 - Analysis of Security Issues for Distributed Storage and Document Management Architectures:**

  This task will analyze vital security elements associated with distributed databases, document storage and management and user based authentication and confidentiality. Of specific focus during this task will be the access control to the specific elements of patent applications stored across a distributed database architecture.

- **Task E3 - Analysis of Security Issues for Distributed Computing Architectures:**

  Task E3 will focus on the security, reliability and availability issues that concern the computer operating system and the Legion computing infrastructure. The security, reliability and availability mechanisms defined in Task 2 are "applications layer" mechanisms, and will rely on the mechanisms provided by the "infrastructure layer" of the DOCT computing complex.

- **Task E4 - Analysis of Security Issues for Network Security Requirements:**

  This task will focus on the conduct of applied R&D on network security requirements for preserving confidentiality and data integrity for legal documents in an electronic commerce environment. Firewall systems, dynamic tracking of operating system integrity, and internal testbed security audits will be investigated.

- **Task E5 - Analysis of Fault Tolerance Mechanisms:**

  Task 5 will result in the development of a DOCT fault tolerance model for the types of response that should be followed given the detection of agent non-completion. This model will identify and characterize parameters which impact the reliability and availability of DOCT. This model will facilitate examination of all components of the DOCT including the communication network, computer hardware and software, and process flow to determine fault tolerance capabilities. The fault model will be based, in part, on the requirements developed under Tasks E-1 through E-3.

- **Task E6 - Integration of Electronic Commerce Security Products into DOCT:**

  This task will focus on the integration of electronic commerce security products into the DOCT architecture with an emphasis on the support of UPSTO electronic filing per the *Electronic Filing Guide.* The security products will consist of a combination of COTS products and development software.

- **Task E7 - Integration of Security Mechanisms into Distributed Architecture:**

  Task 7 will address the technology required to solve the security and fault tolerance issues associated with distributed execution of applications across a distributed architecture. This will focus on the authentication of objects across distributed platforms and the confidentiality and integrity of the results of the distributed program execution. Security intrusions can be viewed as forms of faults. Both intrusion detection software, and fault tolerance systems will be used to demonstrate how to recover from intrusion attempts.

- **Task E8 - Security Monitoring, Audit and Analysis of the DOCT Architecture:**

19

During the last six months of the program, task 6 will be focused on security tests and audits of the demonstrations conducted throughout the test bed. These tests and audits will be conducted from multiple physical locations within the test bed. In addition, test and audits will be conducted from the USPTO location.

F. Develop resource management & load sharing requirements

- **Task F1 - Evaluate resource and load sharing systems:**

  In the primary DOCT architecture, Legion is the core enabling technology that will facilitate the construction and operation of the other software components. Legion will use the AppLeS application level scheduling system that is under development at UCSD. AppLeS will provide scheduling support for tasks that use a small fraction of the resources on a compute platform. Tasks that use a large fraction of the available resources will need to be queued in a batch system. Otherwise they will be unable to access the resources as smaller jobs will always be using the system. The objective of this task is to identify resource and load sharing systems that can be integrated with Legion for possible use in DOCT.

- **Task F2 - Distributed Scheduling:**

  The DOCT project is conducting research on a novel approach for providing scheduling in a distributed environment. Prior attempts have focused on control of all resources by a central scheduler. Instead, DOCT will rely on application level scheduling, in which each application asks the systems for available resources, and then independently decides where the job will get the best service. This approach works well when applications are small in size and use a small fraction of the available resources. The application level scheduling system can then directly schedule small jobs.

- **Task F3 - Integration of queuing system into Legion scheduling system:**

  The DOCT project is conducting research on a novel approach for providing scheduling in a distributed environment. Prior attempts have focused on control of all resources by a central scheduler. Instead, DOCT will rely on application level scheduling, in which each application asks the systems for available resources, and then independently decides where it will get the best service. This approach works well when applications are small in size and use a small fraction of the available resources. When jobs use a large fraction of the available resources, jobs should be queued so that they can be executed one at a time. An interface between the two approaches will be built by providing estimates of the wait time expected for jobs that are submitted to a batch queue. The application level scheduling system can then directly schedule small jobs, and pick the shortest waiting time from the batch queues for larger jobs.

- **Task F4 - Workflow performance testing and analysis:**

  The goal of this task is to develop an heuristic model to describe the performance of the DOCT testbed. The components of this analysis include:

  1. tuning of the scheduler for optimizing turn-around time for selected classes of jobs
  2. identification of bottlenecks that will limit the ability of the system to manage a large number of agents.
  3. analysis of the interactions between the archive, data caches, and the network for the efficiency with which data can be moved
  4. analysis of the amount of time spent on each step of the workflow

20

5. analysis of the associated data flow or movement that results from agent execution

The analysis will be based on a practical engineering estimate of the capabilities of the system. This will involve measuring the characteristics of the system (number of agents in simultaneous use, amount of data being moved, contention within the networks, fraction of resources that the agents require for execution, response time) for a variety of workloads. The results will be analyzed for correlations between load conditions and the measured performance.

G. Develop intelligent software agents

Task G1 - Familiarization with the DOCT Environment

- **Task G1-1 - Determine Agent Framework Requirements in the DOCT Environment:**

   This task will study the architecture and tools that are to be used in DOCT to identify requirements for the software agent framework. The study will include and investigation of API capabilities, integration aspects with other DOCT tools, reliability and failure analysis, DOCT requirements, and other key aspects. From this study, a preliminary set of software agent framework requirements will be assembled.

- **Task G1-2 - Prototype sample software agent framework and agent in DOCT environment using Legion, Open Text, DMS, and database archive:**

   This task will prove that the components in the DOCT system can work together, and that software agents can be used with these components. The agent to be used will most likely be a preexisting agent with functionality that is targeted for testing purposes. This is essentially a proof-of-concept experiment to determine whether the selected architecture for DOCT will work, and allows for changes to the architecture before major software development work begins.

Task G2 - The Software Agent Framework

- **Task G2-1 - Requirements Analysis: Determine USPTO agent applications and develop an agent framework for the DOCT Environment**

   This is the first task for agent framework development, and will involve analysis of patent office documents, data, and processes to identify candidate agent applications. While some agent applications have already been identified for this project, this task will provide a more precise definition of the requirements for each of these agents. In some cases, it may reveal additional agents to be developed, or may remove agents from the current list. This information will be used in development of the underlying agent framework, and a selection of agents from this list will be implemented as part of this task.

   This task will also define the confidentiality and security requirements that should be incorporated into the agent framework. The results from tasks E1-E8 will be used to create a tailored set of requirements for agent applications, including specific security and confidentiality goals that should be met. This may result in additional agents to be developed.

   Finally, fault tolerance issues will be addressed, once the agent types and security requirements have been defined. Since agents will have varying run lengths, security requirements, and other key features that affect the level of fault tolerance required, this task will study the agent framework and agents individually, and determine the fault tolerance needs. Once this has been

21

performed, the fault tolerance requirements will be included in the agent framework as appropriate.

A preliminary list of candidate agent applications has already been compiled, and is listed below. Note that this is a first draft and is subject to significant change, but should provide some idea of the kinds of functions that agents and the framework will be used to perform.

Current agent application candidates include:

- Electronic Filing Registration
- Electronic Filing/Commerce Initiation
- Electronic Document Check-in
- Document Unbundling
- Signature Validation
- Data Integrity (several levels)
- Receipting (several levels)
- Date-time Stamping
- Digital Signature Application
- Data Validation (possibly in Java, for use by applicant)
- Rendering/Viewing Agent (possibly in Java, for use by applicant)
- Missing Parts Identification (possibly in Java, for use by applicant)
- Electronic Filing Date Assignment
- Notification
- EFW Initialization (initialize DMS, extract data elements, a.k.a. PALM)
- Presumptive Classification
- Workflow (monitors, strings other agents together to perform task)
- Search (may involve multiple agents)
- Query/Status Request
- Interference Searching
- Allowances/Rejections (part or an examiner toolkit)
- Office Actions
- Amendment Creation
- Amendment Collation & Processing
- Publication
- Security
- Meta-agent Monitor (for fault tolerance and agent tracking)

- **Task G2-2 - High Level Design: Study current tools and agent frameworks available; design the DOCT agent framework and initial agents:**

  This task focuses on how to build the agent framework and a sample set of agents as described in the requirements document from G2-1. A study of current agent frameworks will be performed, and the capabilities of these will be compared to the requirements of DOCT to determine which framework provides the best match. As DOCT will be pushing the limits of metacomputing capabilities, it is expected that any current framework will require significant augmentation to work, and the high level design produced by this task will define how that will be accomplished.

In addition to agent frameworks, an investigation into knowledge based frameworks will also be performed to support those agents that will require some level of artificial intelligence. The focus of this study will be on identifying current rule-base, frame-based or other knowledge-based systems on which to build the agent intelligence structure.

The next step in this task will be to design the interfaces for the DOCT tools. This task will be significantly simplified by Build 1, which prototypes the tool interfaces. This task will tailor the tool interfaces to support the agent framework requirements identified in G2-1. In addition to DOCT tools, this step will also investigate current COTS tools that could be used to help in the design and implementation process for software agent development. After this investigation, selection of these products will be purchased for the DOCT development environment.

Finally, a sub-set of agents from those identified in G2-1 will be designed for use in the agent framework. This agent group will be selected to provide testing and proof-of-concept capabilities for the agent framework.

Subtask G2-3 - Build 2 Detailed Design/Implementation: Develop agent framework and initial agents based on build 2 designThis task focuses on the development of the agent framework, supporting applications, and the sample set of agents design in task G2-2. Interfaces with Legion, Open Text, the selected document management system, and the DOCT databases will be developed by the respective DOCT entities, while the basic framework will be developed by SAIC. As a result, this task will require significant coordination.

Note that detailed design, implementation, and testing are integrated together in this task. Since this project is only a proof-of-concept, no formal testing or low-level design (i.e. PDL) will be performed for this project. As a result, the applications produced during this phase will be accompanied by a document that describes any lessons learned during development, and future tasks that would need to be performed to complete the applications. This document will help during future updates and production of these applications.

- **Task G2-3 - Build 2 Detailed Design/Implementation: Develop agent framework and initial agents based on build 2 design:**

This task focuses on the development of the agent framework, supporting applications, and the sample set of agents design in task G2-2. Interfaces with Legion, Open Text, the selected document management system, and the DOCT databases will be developed by the respective DOCT entities, while the basic framework will be developed by SAIC. As a result, this task will require significant coordination.

Note that detailed design, implementation, and testing are integrated together in this task. Since this project is only a proof-of-concept, no formal testing or low-level design (i.e. PDL) will be performed for this project. As a result, the applications produced during this phase will be accompanied by a document that describes any lessons learned during development, and future tasks that would need to be performed to complete the applications. This document will help during future updates and production of these applications.

Task G3 Software Agent Development:

- **Task G3-1 - Requirements Analysis/High Level Design: Update requirements and design based on build 2 results, add additional agents:**

This task marks the beginning of the second build of agents and the framework. It will address the lessons learned and future tasks from build 2, and update the framework for those items that are critical to completion of the project. In addition, distributed agent capabilities, work flow capabilities, and additional agents will be implemented in build 3.

This task will review the requirements and design of build 2 and update them where appropriate. In particular, new Legion and Open Text capabilities will most likely be available by this time, and need to be addressed in the requirements and design. Requirements definition and design of the next group of agent capabilities will also be performed.

- **Task G3-2 - Detailed Design/Implementation: Update agent framework and develop new agents:**

    This task will implement the design created in task G3-1, which includes updates to the agent framework for Open Text and Legion, distributed agent capability, and development of new agents. Specific agents to be developed as part of this task include workflow, search, and query agents for USPTO applications. Some or all of these agents will be designed to run in a distributed fashion in the DOCT environment.

    The results of this task will be an updated agent framework and additional agents, as well as a document describing the lessons learned and future tasks relating to the developed applications.

Task G4 - Advanced Agent Applications:

- **Task G4-1 - Requirements Analysis: Study advanced agent applications and determine impact on current agent framework:**

    This task marks the beginning of the second phase of the proposal, and also the beginning of the third build for the agent framework and agents. The tasks are aimed at proof-of-concept for advanced agent applications in the USPTO, and ability for the agent framework to be applied to other government agencies. For the USPTO, three advanced applications will be studied: interference searching, bi-directional document processing between applicants and USPTO examiners, and on-the-fly/on-demand collation of patent documents. To prove the reusability of the agent framework, a study into environmental agent applications with ODU will be performed.

    From these studies, a set of requirements for these advanced agents will be produced, and the current agent framework requirements and design will be reviewed and updated.

- **Task G4-2 - High Level Design: Update design based on build 3 results and advanced requirements, design advanced agents:**

    This task will focus on updating the design of the agent framework and designing the advanced agents identified in task G4-1. As with build 3, any updated functions from Legion or Open Text may also be incorporated into this design.

Task G5 - Detailed Design/Implementation & Final Demonstration: Update agent framework, prototype advanced agents, demonstrate capabilities:

- **Task G5-1 - Build advanced agents for the final demonstration:**

    This task will build the advanced prototype agents and environmental agents to prove that the agent framework is reusable, and prepare for the final demonstration and report for DOCT. Implementation tasks for the advanced software agents will occur as time permits, with primary focus during this task on

24

report and demonstration preparation. Any agents that are not implemented will be documented in the final report.

H. Implement planning & management tools and procedures

- **Task H1 - Develop and Maintain a Detailed Research and Development Plan and Schedule:**
  Develop and maintain a detailed plan, description, and schedule for all activities to be performed under this project.
- **Task H2 - Develop and maintain a "Concept of Operations":**
  The Concept of Operations (CO) will describe the DOCT test bed in terms of it features and capabilities, its relevance to HPCC evolution over the coming decade, and opportunities for other federal agencies to participate.
- **Task H3 - Provide Overall Management & Coordination:**
  Provide overall management and technical coordination support with sponsors, participants, working groups, and the Steering Committee.
- **Task H4 - Provide Scheduled Demonstrations:**
  The DOCT demonstration tasks, as defined in above in the technical task summaries consist of semi-regular briefings and local/remote interactive workstation sessions which demonstrate selected functions and capabilities of the Distributed Object Computational Testbed. For ease of management and presentation, the demonstrations are arranged into 8 Demonstrations Groups, which are identified by theme and time frame (see Section 5).

APPENDIX B: Task Issues

In total, there are 8 Task Groups, which are further subdivided into approximately 50 tasks, or work elements.

A) Develop and document data models

B) Establish metacomputing testbed

C) Develop archival storage and retrieval systems

D) Integrate object computation and data handling systems

E) Define assured service availability & fault tolerance/security requirements

F) Develop resource management & load sharing requirements

G) Develop intelligent software agents

H) Implement planning & management tools and procedures

A. Develop and document data models

- **Task A1 - Analyze Open Systems and Industry Approaches to Represent Complex Work Units:**

    The *Draft Implementation Guide for Electronic Filing* (Chapter 6) identifies a number of R&D issues, which need to be addressed in the development of a comprehensive "roadmap" for representation, interchange, and handling of complex work units in a legal and regulatory framework. Particular issues to be addressed under this task include:

    - What is the availability of well-defined and stable Application Portability Profiles (APPs) for the complex work units?

    - Is the CWU representation mechanism an open, industry, or proprietary standard, and "who" supports it?

    - What is the availability of authoring tools, a reference standard viewer, and other evaluated products?

    - What is the expressive power of the representation mechanism? How "intelligent" is the representation mechanism?

    - What is the potential impact of the representation mechanism on agency, industry, and public information systems?

    - What are the cost-benefit considerations?

    - What is the long-term stability of the representation mechanism?

    - Is the representation mechanism intrinsically searchable?

    - Are there alternative approaches which are preferred, and what is the industry/agency/public demand for new methods of representing CWU objects in an intelligent format?

    - Can the complex work units be rendered for publication purposes?

    - What is the preferred approach for multimedia objects such as MPEG files to represent complex work units?

- Is the particular complex work unit recommendation mechanism sufficiently mature, and what is the level of interest of national and international standards bodies with respect to standardizing on the particular format?

- **Task A2 - Evaluate Hardcopy Scanning, Conversion, and Translation Methods:**

  This task recognizes that for the foreseeable future that the USPTO and other federal agencies will continue to have a requirement for receiving and processing hardcopy, paper-based, scientific and technical documents. As concepts for electronic filing and electronic commerce evolve, there will likely be an increasing requirement to scan and convert paper-based filings to an equivalent intelligent electronic format. Additionally, the USPTO, like many other federal agencies, is confronted with a large backfile conversion problem for complex scientific and technical documents. This task is expected to address a number of technical, managerial, and operational issues associated with the scanning and conversion process including:

  - What is the required optical quality of the input documents, including clean-up such as speckle removal and de-skewing?

  - What is the requirement to constrain the variability of the input using standard forms on the front end?

  - What are the accuracy requirements for conversion?

  - What is the target format of the data stream from the OCR device (e.g., tagged ASCII, display code, RTF, TEX)?

  - How can visual "clues" about the structure and content of the document be preserved within the OCR data stream to facilitate downstream SGML autotagging?

  - What is the preferred approach to handle ISO character sets beyond 7-bit ASCII?

  - What is the interaction of the OCR device with various fonts and publication features (such as kerning, proportional spacing, etc.)?

  - When should the object within the paper-based document be zoned and handled as a graphic; as opposed to being intelligently recognized and converted?

  - What is the preferred approach to recognition of tables, mathematical, and chemical expressions?

  - Can OCR techniques provide recognition of mathematical and chemical expressions beyond comparatively simple in-line expressions containing just subscripts, superscripts, and various ISO characters?

  - What are the primary economic factors associated with scanning and conversion; and how can these factors impact future incentives to support an alternative electronic filing approach?

  - When is it necessary to re-key and/or re-draw complex work units (as opposed to trying automated OCR techniques)?

- **Task A3 - Explore SGML Autotagging Technology:**

  The *Draft Implementation Guide for Electronic Filing* (Chapter 2) identifies a number of SGML Document Type Definitions (DTDs) to support the bi-directional electronic filing concept. The DTDs define the required structure and content of the electronically-submitted documents. In support of continued paper-based submissions, the USPTO has also developed a series of standard forms to help structure the

27

documents with a goal of streamlining downstream processes for scanning and conversion. For the most part, the information required in the electronic filing DTDs mirrors the information required in the equivalent paper-based standard forms. Thus, it should be possible to scan and OCR the standard forms and SGML autotag the documents. A similar statement can be made for published backfile patent documents by utilizing the WIPO ST.32 DTD. (Note: The feasibility of large-scale backfile conversion to SGML-tagged ASCII has not yet been successfully demonstrated, and some changes may be necessary to the ST.32 DTD to facilitate the autotagging process).

In any SGML conversion process, complexity arises, however, with the requirement to handle: complex mathematical and scientific expressions, references to external objects such as multimedia objects and graphics files, references to text passages in other external documents, and internal references and potential hyperlinks such as claim references. Within this task, particular issues to be addressed will include:

- What is the requirement to constrain the variability of the input using standard forms on the front end? Note: published backfile patent documents are already constrained.

- What are the accuracy requirements for SGML autotagging?

- What tools and techniques are available to support SGML autotagging?

- What is the potential impact of complex vs. relatively simple DTDs (i.e., an SGML-Lite approach)?

- Should documents be converted to HTML or SGML?

- What is preferred target format of the data stream from the OCR device (e.g., tagged ASCII, word processor display code, RTF, TEX)?

- How can visual "clues" about the structure and content of the document be utilized to facilitate downstream SGML autotagging?

- How can various fonts and publication features (such as kerning, proportional spacing, etc.) best be used in support of autotagging?

- What is the preferred approach to zone and autotag external entity references such as graphics and multimedia objects?

- What is the preferred approach to SGML autotagging of tables, mathematical, and chemical expressions?

- What is the preferred approach to identify and autotag internal references for hyperlinking purposes?

- What are the primary economic factors associated with SGML autotagging; and how can these factors impact future incentives to support alternative electronic filing?

- What are the cost-benefit considerations of SGML autotagging?

- What is the requirement to perform validation parsing of the SGML-tagged data stream?

- What happens if the SGML autotagging process fails? or if there are errors?

- What are the legal and regulatory implications of scanning, conversion, and SGML autotagging given that the converted SGML-autotagged document is "different" from what the applicant originally submitted?

- What is the potential impact of SGML autotagging technology on future information systems within the USPTO and other federal agencies?

- What are the lessons learned and experiences of SGML autotagging in the Microsoft Patent Workbench, and how can they be best applied in this project?

- **Task A4 - Develop and Evaluate a STEP-Based Approach to Representation of Intellectual Property:**

  The *Draft Implementation Guide for Electronic Filing* currently has a placeholder for future incorporation of ISO STEP to interchange product and process model data as part of a patent application. A number of technical, operational, and managerial issues need to be addressed to determine the feasibility of actually using STEP. Within the DOCT project, particular issues to be addressed as part of this task include:

  - What is the availability of well-defined and stable Application Protocols (APs) for STEP and how can they best be adapted to support the patent application process?

  - What additional new capabilities would an ISO STEP-based approach to electronic filing afford to the patent applicant community and the PTO?

  - What is the availability of authoring tools, a reference standard viewer, and other evaluated products for an ISO STEP approach?

  - What is the expressive power of the ISO STEP representation mechanism?

  - What is the potential impact of STEP on PTO, industry, and public information systems?

  - What are the cost-benefit considerations of a STEP approach?

  - What is the preferred approach to incorporating and/or embedding SGML-tagged documents into the STEP file(s)?

  - What is the preferred approach to incorporate and/or embed patent-unique data elements (such as bibliographic elements and claims) into STEP-based approach?

  - To what extent can a STEP-based approach to representation of an invention support the legal objective of full design disclosure and disclosure of the best manufacturing process?

  - What industries would find a STEP-based approach to be most interesting and useful?

  - What is the preferred approach to incorporate graphics and other objects into the STEP files and how well can STEP files be translated into 3-D VRML representations for visualization purposes?

  - Insofar as STEP can represent sufficient features to provide for the represented product or process to be simulated, would such a simulation be sufficient as proof that a claimed behavior of a invention is really achieved? Conversely, could such as simulation wrongly convince an examiner?

- **Task A5 - Develop a Validation Mechanism to Support Electronic Filing:**

The *Draft Implementation Guide for Electronic Filing* (Chapter 8) identifies a significant number of R&D issues, which need to be addressed in the development of a Validation Mechanism to support patent application management in the international legal and regulatory environment of the USPTO. Particular issues to be addressed include:

- How should the Validation Mechanism be structured and really work?

- What is the proper role of open, industry, or proprietary standards within the context of a Validation Mechanism?

- What is the potential availability of an accepted Reference Standard Viewers, and other evaluated products? How should this information be disseminated to the patent applicant community?

- What is the preferred role of commercial-off-the-shelf (COTS) products and can they be specified to be "the" Reference Standard Viewer for explicit file types in an international legal and regulatory environment?

- Can reasonably effective mechanisms for validation of documents be defined in the context of "real use" and "real economic risks"?

- Can a reasonably effective Validation Mechanism be demonstrated in an Internet-based, real-time, on-line electronic filing environment?

- Can Application Portability Profiles (APPs) for allowable file types be relatively stable over long periods of time?

- What happens to Reference Standard Viewers and evaluated products as hardware and operating systems environments change over periods of time? Must they also be archived?

- For reference standard viewers and evaluated products, what are the legal and rules implications with respect to the actual presentation or visual rendering of the object (i.e., graphic, text, multimedia object, complex work unit)? Must it look the "same" and/or sound the "same" to both the sender and receiver of the information? What is the rules impact of differences in hardware and/or software suites on the physical rendering of the object?

- In an international patent application environment, what are the respective responsibilities of the sender and receiver of the information within the concept of a Validation Mechanism?

Tasks A6-A8 - VRML

- **Task A6 - Develop VRML Submission Support :**
  VRML content is typically made up of multiple files, linked together via URLs embedded within the files. These URLs may reference one or more of the following file types:

  - VRML files
  - VRML external prototype files
  - JPEG, PNG, or GIF image files
  - MPEG movie files
  - WAV or MIDI sound files
  - Java or JavaScript program script files

30

In this context, and as indicated in the *Draft Implementation Guide for Electronic Filing*, a future patent (or trademark) application may include a VRML model of a gadget (or a trademark). That gadget may be built from multiple component parts, each in its own VRML file or external prototype file. Those gadget components may use images as textures to add detail, like adding a decal to a model airplane. MPEG movie files may be used to create moving imagery within the gadget. WAV and MIDI files may implement essential sound aspects of the gadget. Finally, animation of the gadget may use one or more Java or JavaScript program fragments.

Submission of VRML content requires that each of the component files of the content be identified, retrieved via their respective URLs, and bundled together in a manner suitable for archival storage and retrieval. Once fully retrieved and bundled, all component files should be checked for validity. Trusted parsing software will be purchased or developed that confirms the validity of the syntax used in all VRML files, and confirms the validity of all JPEG, PNG, GIF, MPEG, WAV, MIDI, Java, and JavaScript files submitted as part of the VRML submission.

To assist in validity checking, the submission process will define what is valid. Within the *Draft Implementation Guide*, this process is referred to as the Validation Mechanism. For complex VRML files, validation needs to consider the full scope of embedded standards including: VRML, JPEG, PNG, GIF, MPEG, WAV, MIDI, Java, and JavaScript.

The issues addressed, then, include identifying the steps involved in the submission of VRML content, the development of retrieval and bundling software, the development of validity definitions, and the development of software to determine the validity of VRML, JPEG, PNG, GIF, MPEG, WAV, MIDI, Java, and JavaScript components included in a VRML submission to a document archive.

- **Task A7 - Examine VRML Retrieval :**
  The design of VRML enables the generic description of shapes and scenery for purposes ranging from shape databases to virtual reality entertainment. To address such a broad charter, VRML provides the author a large number of alternative approaches to creating the same "thing". This increases the power of the author, but significantly reduces the ability of generic 3-D content-based retrieval algorithms to recognize semantic structures within a VRML file. A sphere shape, for instance, can be built in a nearly infinite variety of ways in VRML. A trademark search looking for logos based around a sphere would be hard-pressed to generically recognize sphere shapes given such a diverse range of sphere building approaches in VRML. Searches based upon patent or environmental data set features will encounter the same problems.

  The principal issue, then, is whether or not 3-D content-based retrieval of archived VRML content is possible, and if so in what time frame (i.e., today?, within a few years?, within a few decades?, etc.). Preliminary investigations on this topic indicate that the problem is an order of magnitude more difficult than similar work in-progress for the last decade on 2-D image content-based retrieval.

A major issue to be addressed under this task to examine and determine whether alternative 3-D shape file formats may provide a better chance for content-based retrieval, while still retaining the ability to describe arbitrary shapes, colors, textures, sounds, and animations.

- **Task A8 - Examine VRML Display:**

The 3-D computer graphics industry is less than three decades old and has, only recently, reached a level of interactivity and widespread use that makes archival storage of 3-D content a reasonable topic for discussion. During the last few decades, 3-D graphics performance has increased along with computer processing performance at an average rate of about a factor of 2 increase every year. Significant jumps in that performance rate occur each time new graphics algorithms become available. Such a jump occurred about 10 years ago with the advent of raster graphics pipelines, and is in progress today with the incorporation of 2-D imaging techniques in Microsoft's forthcoming Talisman graphics hardware.

Along with the development of new graphics hardware and algorithms, new graphics standards have been developed, including CGM, GKS, PHIGS, IGES, STEP, and now VRML, plus a wide range of industry *de-facto* standards. Across industry, older standards die quickly as new standards arrive that can describe the newest features of graphics hardware and algorithms of the day. This fast pace of graphics technology creates a high turn-over rate in graphics standards that will, ultimately, affect VRML, or any other 3-D graphics file format.

The principal issue to be addressed in this task is whether any 3-D graphics file format, VRML or otherwise, can meet the long-term archival needs of the Patent and Trademark Office, other federal agencies, the Trilateral Offices, the National Archives and Records Administration (NARA), industry (as a whole), and the requirements of commercial archival scientific databases. As a practical matter, it is unlikely that any single format can survive more than 3-5 years in today's rapidly advancing graphics market. Even those formats that survive this length of time typically undergo many revisions in order to keep up with technology. This creates a serious problem for long-term archival storage, retrieval, and display/rendering and leads to the possibility of orphaned data that uses out-of-date formats or format versions and can no longer be displayed/rendered a few years after it is archived.

A secondary issue to be addressed is what to archive, along with data, in order to ensure that the data can be rendered and displayed again in the future. If, for instance, the data is VRML content, then VRML display and rendering software would be an appropriate candidate for archival storage. Unfortunately, that software uses underlying software (such as an operating system) provided by assorted computer platform vendors. That software may also need to be archived. Unfortunately, that software uses underlying hardware, which also should be archived. This large set of archived support hardware and software becomes cumbersome and impractical, leaving the original problem of what to do to ensure that future users of the archive can retrieve, display, and otherwise, render, the archived VRML content.

Tasks A9-A11: Environmental Data

- **Task A9 - Model and Represent Environmental Data:**

The key issues to be addressed in this task include:

- What is the preferred approach to tracking of documents and data sets?
- What is the preferred approach as well as requirements for text and image search for normal text and graphics files?
- What is the preferred approach for demonstrating an ability to perform content-based searching of documents and data sets?
- What is the preferred approach to present search results for the environmental documents and data sets?
- What is the preferred approach to hyperlinking between and among environmental documents and data sets?
- What is the preferred approach to represent and visualize multimedia environmental documents? (e.g., VRML?, SGML?, or Other?)
- To what extent can the approaches for representing, visualizing, and searching environmental documents and data sets be generalized to other classes of documents and data in other scientific disciplines?
- Can the USPTO common content model for SGML contained within the *Draft Implementation Guide for Electronic Filing* be broadly applied to environmental documents?

- **Task A10 - Integrate Environmental Database with Archive:**
  The key issues to be addressed in this task include:
  - Multimedia documents (VRML, SGML, other)
  - Search by content
  - Metadata development
  - Database with archive integration
- **Task A11 - Integrate Environmental Database with Legion:**
  - Multimedia documents (VRML, SGML, other)
  - Database with archive integration
- **Task A12 - Legacy Data Migration and Data Load Module:**
  It is understood that migration of legacy Messenger text and CSIR image data to the advanced DOCT architecture is a major item of interest to the USPTO. It is also known that a major limitation of the migration of the existing Messenger legacy text data is that the published patent document can not be re-composed from the text stream. This is partly because external callouts to separate graphics files (such as chemical Markush structures) are not within the data stream. This is another way of saying that the Messenger text data stream does not represent a "true, mixed-mode, compound document". Under this subtask, the particular issues to be addressed include:

  - What is the requirement to translate the Messenger Green-book tagged data to an SGML format, and if so, to which SGML Document Type Definition (DTD)? Do efforts to address this issue need to consider the known capabilities and limitations of the current version of the WIPO ST.32 DTD? Alternatively, can an SGML DTD be inferred from the structure of the Green Book tags and would it be better to use this new DTD as opposed to the ST.32 DTD?
  - Can/Should the Messenger data be directly indexed as tagged ASCII by Open Text Livelink Search (as opposed to translating it to SGML) and what pre-processing is required?

33

- What are the requirements and concerns about exception processing if there are errors in the Messenger data stream?
- What is the preferred approach and methodology for translating unique and proprietary Messenger character codes to an open ISO character set equivalent? Also, what is the preferred approach to handle subscripts, superscripts, bolding, italics, underlining?
- What are the downstream user interface requirements to enter, display, and search on ISO characters or on proprietary Messenger Green Book characters?
- What are the requirements and preferred approach to handle hyphenation at the end of a line in the Messenger data stream for pre-processing, translation, and indexing purposes?
- What are the issues, if any, associated with maintaining page and column integrity when importing, translating, pre-processing, and indexing the Messenger data stream?
- What are the major concerns and issues in handling tables within the Messenger data stream? What about format of the tables? How should tables be indexed, stored, searched, retrieved, and visualized?
- What are the potential legal implications, if any, of translating and converting the legacy data (both Messenger text and CSIR images) to any new format?
- What are the downstream business considerations of any conversion, translation, or pre-processing of the legacy data?
- From a data loading and data architecture perspective, should the text in the Messenger files be broken up into separate files with a separate file for each patent document? If so, what directory structure should be employed? Note: a similar issue needs to be addressed for patent images in Yellow Book format. Should the image files be broken up into separate page image files for each page within each separate patent document? How should text data be hyperlinked to the image data?
- What are the PTO's requirements, goals and objectives for incorporating any legacy trademarks data into DOCT? (Note: It is understood that the PTO wants to incorporate both patent and trademarks data into the DOCT environment.)
- What are the logistics considerations behind receiving the legacy patent and trademark data? (e.g., how much needs to be sent? format?, media?, schedule of delivery?, requirements for receipt and access control?, etc.)
- What are the basic requirements for replicating and distributing the legacy text and image data throughout the distributed DOCT heterogeneous databases? What is the plan and concept for replication and archival storage and retrieval to meet the DOCT R&D objectives?
- **Task A13 - Prototype the *Draft Implementation Guide for Electronic Filing* with Intellectual Property Community Participation:**
  Under this subtask, the major issues to be addressed include:
  - What is the adequacy and feasibility of the Implementation Guide as a technical basis for defining processes for bi-directional electronic filing with amendment processing in a secure electronic commerce environment using the Internet?

- What is the required skill level of the patent applicant community to author and receive mixed-mode, compound, SGML-compliant documents containing complex work units (CWUs), graphics, and multimedia objects?
- What COTS products are preferred for authoring, rendering, and validating documents and files in an electronic filing environment? Are the products adequate?
- What is the level of interest of COTS vendors to tailor their products to the Implementation Guide?
- What is the feasibility of the patent applicant community in interacting with a distributed, fine-grained, Web-based, enterprise-wide, document management system (DMS) to electronically check-in applications, to provide search support, to track the status of the application using a workflow subsystem within the DMS, to provide notification, and to manage the patent and trademark prosecution process (including amendment processing at the element level)?
- What is the adequacy of electronic commerce security within the Implementation Guide including: digital signature, secure hashing, secure date-time stamping, and encryption.
- Is the concept of a Validation Mechanism sound and workable?
- Lastly, what changes and modifications are required for the Implementation Guide and what are proposed rules changes?

- **Task A14 - Document Classification using Self-Organizing Maps:**

  The following Research Issues will be addressed:

  - Implementation of classification on large numbers of textual documents: many of these algorithms have not been tried on data sets as large as the patent collection.

  - Selection of classification algorithms: a wide variety of algorithms including neural networks, genetic algorithms, and statistical methods are available. Different algorithms may result in different classifications

  - Automatically generated classifications vs. PTO standard classifications: current work involves the automatic selection of subject areas, we may be able to adapt this to the standard classifications.

  - Integration with DOCT: there my be many difficulties in tightly integrating the classification system into the DOCT architecture.

B. Establish metacomputing testbed

- **Task B1 - Install Testbed Infrastructure:**

  There are no research issues that need to be addressed as part of this task. The only issues here are related to logistics. Each site has personnel who will coordinate the infrastructure installation at that site. For software that is installed at several sites, the site personnel will coordinate with the personnel responsible for that software system.

- **Task B2 - Installation, Development, and Integration of Livelink Search:**

  The major issues for migration of the legacy Messenger text database to the Open Text Livelink Search environment are discussed in RDPS A12. This subtask is largely

concerned with initial installation and upgrades to Livelink Search. Over the period of performance of the DOCT contract, Open Text will be making enhancements and upgrades to Livelink Search. Under this subtask, the only real issues to be addressed are:

- What is the anticipated nature of the planned enhancements?
- When will the planned enhancements be installed within DOCT?

Insofar as the planned enhancements are being independently funded by Open Text and are being made to a commercially marketed product, discussion of the planned enhancements and their anticipated schedule for deployment will be discussed on a non-disclosure basis with the USPTO.

- **Task B3 - Install and Update Legion System:**
  This task only deals with the installation of the Legion software. The issues to be addressed include the coordination of the installation of Legion upgrades across the various sites/platforms of the DOCT testbed. For example, installing an upgrade may require all relevant sites to coordinate and temporarily shut down the Legion system while the upgrade is being done. On the other hand, depending on the design of the software, it may be possible for a "down-level" version of Legion in one domain to interact with an upgraded version of Legion running in a different domain/jurisdiction.

  Another issue is related to the porting of a given version of Legion to platforms in the DOCT testbed that are not currently supported, e.g. the Cray T3E or Intel Paragon.

- **Task B4 - Analysis of Alternate Document Management Systems:**
  In coordination with the USPTO, the following have been identified as the major issues (and evaluation factors) that need to be considered in selecting an enterprise-wide document management system (DMS) for integration into the DOCT environment:

  - Does the architecture support management of documents at the element or component level (i.e., fine-grained management)?
  - Does the architecture provide support for document management at any or all of the following levels?
    - Image-based or "BLOB" (i.e., binary large object) document
    - Mixed-mode intelligent word processing file, graphics file, or media object file (i.e., where all of the components of the document are contained within a unitary file)?
    - Mixed-mode, compound document approach with or without support for SGML (i.e., text, graphics, media, and character set objects are treated and managed as separate entities)?
    - True, mixed-mode, compound document approach with the document managed at the SGML element/entity level (i.e., "fine-grained" document management)?
  - Does the architecture provide for handling of an "arbitrary" Document Type Definition (DTD)
  - Is the architecture scaleable?

- Is the product stable and reliable?
- Is the product "workflow enabled"? What workflow features are supported?
- What is the model or paradigm for version control?
- To what extent does the DMS provide a built-in capability to audit all changes and modifications to a document or its elements? Is this audit trail ability robust enough for legal and regulatory purposes?
- For fine-grained document management systems, can elements (or objects) be shared across multiple documents?
- To what extent does the DMS provide support for a CALS Modification Request (MODREQ) capability, which can support amendment processing?
- To what extent is text and image search fully integrated with the document management system (i.e., do not want separate indices)?
- What is the overall level of integration of the product (i.e., with text search, with workflow, with printing and publishing, with viewers, with an RDBMS (especially Oracle), with archives, with authoring tools, with customized tools through an API, etc.)?
- What open systems standards and industry standards does the DMS support?
- Is the user interface flexible and extensible?
- Does the product have a robust application programming interface (API)? In particular, does the API provide a capability to intercept data and procedure calls between the applications software and its database thus allowing future integration with heterogeneous databases (including Oracle, the USPTO's standard RDBMS)? Also, does the API support tailorable user front ends which can be integrated with software agent technology?
- What is the cost per server? per seat?
- Does the product provide support for distributed-group document creation, management, and use? How does the document management system manage the process within a group?
- Is the product object-oriented?
- Does the product provide support an electronic folder paradigm?
- Does the product provide for distributed low-cost document management system clients across the Internet/Intranets (i.e., DMS clients running on Web browsers)?
- Does the product have a scripting capability to generate forms for data capture purposes playable across the Internet on a Web browser (i.e., CGI capability)?
- Does the document management system provide for electronic commerce security features including:
    - Originator authentication (i.e., digital signature)
    - Data and message integrity (i.e., secure hash algorithm)
    - Confidentiality (i.e., encryption)
    - Non-repudiation
    - Assured service availability (through secure digital date-time stamping service)
- Does the DMS provide the security and locking features at the element level? What processes and/or standards are involved in element-level security? How robust is the security mechanism at the element level (e.g., Does it use digital

37

signature and secure hashing or merely password protection?) Can certain elements be denied for access to individuals and classes of individuals?

- To what extent does the DMS provide for Internet-based electronic commerce security features such as secure sockets layer (SSL), secure hypertext transfer protocol (S/HTTP), secure financial transaction protocol, etc.?
- Does the DMS provide any built-in support for imaging and rendering the documents, especially tables, mathematical, and chemical expressions?
- Are there any built-in biases against a "pageless" oriented approach to document management?
- Does the DMS flexibly support both "pageless" and "page-oriented" models of documents?

- **Task B5 - Experiment with Existing Security Options:**

    The following Research Issues will be addressed:

    - Confidentiality and intrusion detection
    - Originator authentication
    - Secure date/time stamping

- **Task B6 - Install Heterogeneous Database Environment:**

    This task deals only with installing the various commercial DBMSs. We will install the latest available versions, and software upgrades will be provided as a matter of course by the individual vendors. As part of the installation, we will provide a demo that shows that data can be retrieved from each one of these DBMS's by a given application. This task does not address the issue of combining data across different database systems, e.g. performing a "join" operation across databases.

- **Task B7 - Load Patent and Trademark Data:**

    - Logistics. Since the amount of data from the USPTO is potentially very large, we need to pay attention to the logistics of receiving the data in its physical medium (tapes), storing it or parts of it at one or more DOCT site, and scheduling the necessary time at the archive facility to load this data. This requires negotiating schedules, space requirements, and other resources needed to do the manual task. It is also necessary to work out the logistics of transforming and indexing the "raw" data received from the USPTO. This can be a compute intensive as well as a storage intensive task.

    - Data Load Module. The expectation is that the USPTO legacy data will be transformed and indexes into a format that is more useful and efficient for fast searching. Since the original data is expected to be very large, one can expect the transformed data also to be large. Thus, it may need to be archived as well. In order to plan for adequate archival resources, it will be necessary to have an understanding of the data load module which will describe the required transformation and indexing for the legacy data. Conversely, the fact that the entire data set is very large and needs to be archived may influence the data load module as well.

    - Actual processing/loading of data. Since the legacy data set can be very large, the actual loading of the data itself becomes an issue since it can be a long-running task. Once the size of the data set is known, it will be necessary to evaluate alternative methods for loading it into the archive, estimate the total amount of time and resources and risks involved in each case, and then choose the best approach.

38

Similarly, it is also necessary to estimate the computational and storage resources needed by the indexing step and to plan for that.

- Confidentiality. At every step of the process, starting from the transfer of data/tapes from the USPTO, to their receipt at the archival site, to storing them in the archives, it is necessary to ensure that the confidentiality of the data is always strictly maintained. This is an essential requirement and will require some attention.

- **Task B8 - Install Archival Systems:**

    Since there will be two archival sites within the DOCT testbed, at SDSC and Caltech, both running HPSS, it is necessary to ensure that the versions of the archival software systems installed at both sites are compatible. In additional, any other support software that is used at one site should also be installed at the other site. In general, this is the issue of ensuring that software installed in a distributed system is kept in synchronization or, at least, is compatible across sites. The same issue is relevant in several other tasks as well.

- **Task B9 - Develop Sample Test Agent:**

    For each DOCT resource that will be accessed by the sample agent, we need to determine how the agent can determine that the resource is accessible and available to be used.

C. Develop archival storage and retrieval systems

- **Task C1 - Develop data storage architecture:**

    - Design of data formats for storing data in the archives. We need to ensure that the formats are all-inclusive so that there is no need to migrate data to new formats in future. For very large data sets, this can be a time consuming operation.

    - Design of metadata related to archival data. Storing appropriate metadata for each archival data set can greatly help in reducing search/retrieval times for the archive. A class of queries could be answered by accessing only the metadata, without going to the original data set. For others, the metadata could be used to reduce the set of possible data sets that need to be accessed.

    - Specifying replication requirements. The actual design of replication strategies will be done under Task C3. This task will identify the requirements for replication, based on the characteristics of the application.

- **Task C2 - Integrate Archival Storage Systems with DBMS:**

    There are various ways in which a DBMS could be integrated with an archival storage system. One is to provide user-defined functions that map the file I/O interface to the archival I/O. When the UDF is invoked, it performs the archive I/O and sends data back to the applications. Another is to modify the DBMS software such that the I/O to the archive is performed internally by the DBMS engine itself. Various possibilities will be explored.

- **Task C3 - Experiment with Replication Alternatives:**

    - Replication strategy. We need to evaluate what the best strategy for replication is for USPTO applications. As mentioned above, there can be a variety of replication strategies ranging from "immediate" to "lazy" replication.

    - It may not be possible to replicate the entire legacy patent and trademark data from USPTO initially. This can be a large amount of data and the resources needed to replicate may be large.

- While we may identify several replication strategies, we may be demonstrate only one or two of them in the course of this project.
- **Task C4 - Use MDAS API's:**
  - We have to judiciously pick DOCT components and data sets for representation as MDAS resources and data sets, so that we can demonstrate the benefits and capabilities of MDAS while at the same time keeping control on the effort involved in doing this.
  - The issue raised above also applies to how the security and scheduling work is handled. We will need to pick a representative set of tasks for demonstration within MDAS, in order to keep the effort in control.
- **Task C5 - Integrate Extended MDAS Capabilities Into Legion:**
  - MDAS will support MPI IO which is an evolving standard for parallel I/O. One issue is whether this standard has the necessary functionality to support DOCT applications.

  - The use of MPI IO provides parallel I/O into an application. It may also be possible to use third party transfer techniques to improve I/O bandwidth in some cases.

  - Sequential applications will need to modified to exploit parallel I/O capability. We will need to examine which level of the DOCT architecture is effected by this requirement.

  - Remote access will require support for authentication mechanisms to validate the access.

D. Integrate object computation and data handling systems

- **Task D1 - Legion Integration with Databases:**

    The integration effort needs to handle mapping of names to a storage address, scheduling of access to the data sets, and should provide support for the wide variety of databases implemented in the DOCT testbed. Particular issues to be addressed are:
    - What database schema can be used to identify a data set? Will the Legion LOID need to be stored as metadata associated with every data set?
    - Will the metadata stored in the database constitute an object that can be manipulated independently of the data set that the metadata describes?
    - Will databases be used primarily as object indexers to data stored in archives?
    - How can levels of address indirection supported by databases be handled? The database may store a pointer to the actual data set, requiring an interaction with Legion to return the true data set location.
    - Is there a common way to handle the multiplicity of databases present in DOCT? These include relational databases which provide a pointer to external files that may be on local disk or in a remote archive, object-relational databases that reference data sets through an internal large object pointer, and object oriented databases that also directly reference large objects.
- **Task D2 - Legion Integration with archival storage:**

    The integration effort should handle mapping of names to a storage address, scheduling of access to the data sets, and should provide support for third party retrieval of data sets. Particular issues to be addressed are:

- What mechanism will be used to force the caching of data within the HSM on disk as opposed to tape?
- How can the use of parallel I/O streams be supported with Legion? This is particularly important for large data sets.
- How can third party transfers be handled? In particular, the data may be moved from network attached peripherals controlled by the HSM directly to the compute platform on which the data access was requested, bypassing the data server platform which is executing the HSM. Thus the data may be returned from a different address than the location where the request was made.
- Should it be possible to reference a data set both through a database front end to the HSM and directly from the HSM? Or should different versions of the data set be defined for the two different access mechanisms?

- **Task D3 - Integrate Livelink Search with Legion:**
  - What level of performance is required in terms of frequency of access? Is the access dominated by the time to read the data, or is it dominated by the overhead time associated with identifying a data set?

  - How can consistency be maintained? Approved patents may be stored in the database or the archive, without the search index being updated. Should updates to store approved patents trigger the generation of a new search index?

  - How can compound documents be adequately supported? For best retrieval performance, all components of a compound document should be archived together as a family. On the other hand, it may be necessary to only index the claims section of a patent, thus requiring access to only part of the patent.

- **Task D4 - Integrate Document Management System with Legion:**

  The issues are substantially the same as in task D3
  - Will it be possible to provide an interface that replace the DBMS API used by the DMS, with an API that provides the same call functionality but references Legion LOIDs? The concern is that the DMS may use a database function that does not preserve persistent states.
  - Will the performance of the interface be adequate for the level of granularity that is expected by the DMS?
  - Will it be possible to extend the DMS interface to handle compound documents that are stored in an archive? This may require explicit commands to cache data on disk to gain adequate performance.
  - Will it be possible to implement a database schema that mimics the document organization used by the DMS? The DMS organization schema may conflict with the organization schema needed for supporting the search engine. If so, multiple copies of the data may have to be stored.
  - What level of performance is required in terms of frequency of access? Is the access dominated by the time to read the data, or is it dominated by the overhead time associated with identifying a data set?
  - How can compound documents be adequately supported? For best retrieval performance, all components of a compound document should be archived together as a family. On the other hand, it may be necessary to only index the claims section of a patent, thus requiring access to only part of the patent.

- **Task D5 - Develop architecture analysis:**

41

The target architecture will affect all issues that involve inter-tool support, as the architecture will define how those tools will interact. The following issues will most likely require inter-tool support, and therefore will be affected by the selection of a target COTS architecture:

- Which COTS text search systems provide APIs that will allow their integration with Legion?
- Which COTS systems provide image search capabilities that can be integrated with Legion?
- Are their existing systems for displaying search results?
- Can document rendering systems be found that interoperate with DBMS and archives?
- Is support for multimedia documents available that can be integrated with the intelligent agent framework?
- Is there a standard intelligent agent framework available that will simplify development of intelligent agents?
- Should name management difficulties be minimized by use of a common file system across all data vaults and compute servers?
- Can task scheduling be adequately managed through use of a global queuing system?
- Should fault tolerance be implemented at the agent level, or in the communications level of the Legion system?
- Is there an emerging standard for document handling APIs that should form the basis for the intelligent agents?
- Is use of a single distributed database mandated for adequate persistent object support, or can databases be federated?
- Can an intrusion analysis system adequately track attempts to compromise operating systems? If this is not feasible, then public access should not be provided into the system.
- What is the minimum degree of partitioning and replication needed to assure availability?
- Is it feasible to log all transactions made on every document, including access patterns?
- Can access be restricted for documents that are being processed, while within the same system public access is provided to approved documents? Will the publication of information have to be segregated from the processing of documents in progress?
- Is versioning of documents sufficient to maintain persistent state?
- Can a system be designed which will recover its state, no matter what the failure mode is?
- Can data integrity be guaranteed without having to resort to read-only hardware systems?

E. Define assured service availability & fault tolerance/security requirements

- **Task E1 - Analysis of Security Issues for Electronic Commerce Applications:**
  1. Confidentiality of documents and objects across a distributed architecture
  2. Intrusion detection of attacks against electronic commerce applications

3. Authentication of document and data originators
4. Non-repudiation of document and data originators and document and data recipients
5. Data integrity of all data in transmission, in storage and during processing
6. Auditability of documents and files during creation, transmission, receipt and processing
7. Secure date/time stamping of documents, files and objects

- **Task E2 - Analysis of Security Issues for Distributed Storage and Document Management Architectures:**
   1. A strong security model for an integrated DOCT architecture
   2. Management of versions of documents, files and object elements through the complete document management process
   3. Management of access control across the distributed architecture for documents, files and objects

- **Task E3 - Analysis of Security Issues for Distributed Computing Architectures:**
   1. Intrusion detection of attacks at all levels of a distributed architecture
   2. Fault tolerance requirements for each level the distributed architecture
   3. Data integrity of all data in transmission, in storage and during processing
   4. Auditability objects
   5. Secure date/time stamping of objects
   6. Management of access control across the distributed architecture for objects and distributed applications

- **Task E4 - Analysis of Security Issues for Network Security Requirements:**
   1. Data integrity of all data in transmission and during distributed processing.
   2. Auditability of objects during transmission and receipt across the network.
   3. Secure date/time stamping of data objects across the network.

- **Task E5 - Analysis of Fault Tolerance Mechanisms:**
   1. Coordination of the initial development of availability requirements between tasks in Section E and initial activities in Section E-5
   2. How can mis-diagnose of faults be handled?
   3. Which recovery methods are suitable for intelligent agents? Re-execution of an agent may not be possible.
   4. Will the initial schedule and level of activity proposed in this task be sufficient to meet the objectives?

- **Task E6 - Integration of Electronic Commerce Security Products into DOCT:**
   1. Data integrity of all data in transmission, in storage and during processing
   2. Auditability of documents and files during creation, transmission, receipt and processing
   3. Secure date/time stamping of documents and files
   4. Integration with the underlying security mechanisms provided at the operating system level

- **Task E7 - Integration of Security Mechanisms into Distributed Architecture :**
   1. Intrusion detection of attacks at all levels of a distributed architecture
   2. Fault tolerance at each level of the distributed architecture
   3. Data integrity of all data in transmission, in storage and during processing
   4. Auditability of objects during processing

43

- **Task E8 - Security Monitoring, Audit and Analysis of the DOCT Architecture:**
  - Can the intrusion analysis software detect attacks sufficiently rapidly that intruders can be disabled before they compromise data?
  - Given an intrusion, can fault tolerance mechanisms provide reliable approaches for rebuilding the data environment?
  - Given that security can not be demonstrated in complex systems because of the complex data interaction mechanisms, can auditing/logging of all operations on data sets be used to identify when data has been compromised?
  - Can replication of data in a distributed environment be used to both safeguard data and detect when data is being corrupted?

F. Develop resource management & load sharing requirements

- **Task F1 - Evaluate resource and load sharing systems:**
  1. How can local scheduling policies on individual compute platforms interoperate with the global Legion scheduling policies? Legion provides mechanisms for load leveling (process distribution and throttling), process creation, distributed (currently homogeneous) inter-process communication, and resource usage tracking (accounting). These mechanisms may require that resources be dedicated to either local use or for use through the Legion environment. Note alternative systems provide a subset of the Legion capabilities.
  2. What communication protocol should be provided for supporting communication for jobs scheduled to execute on multiple compute platforms? Condor, for example, (http://www.cs.wisc.edu/condor) is designed for load-leveling between workstations, and perhaps could be used in combination with PVM, DCE Remote Procedure Calls, Sun RPC, or just TCP sockets, for distributed communication.
  3. Which capabilities required by DOCT are currently missing from Legion? In using the Legion system as-is we'll be able to meet some of the DOCT requirements but not all. We'll prioritize the importance of needed features and develop alternatives where possible. This will include an evaluation of trade-offs and costs.
  4. Which commercial systems could be used for supporting job queuing? Other queueing system possibilities are NQE (the new CRI batch system), Loadleveler, and LSF V2.1 (Load Sharing and Distributed Batch Software, which interoperates with Cray NQS) which are planned to be used in the Accelerated Strategic Computing Initiative project.
  5. What minimum functionality would allow DOCT to proceed? A final "load-leveler" to consider is the NULL-set (or almost NULL). That is, we could we use a simple heuristic of mapping task characteristics (or task types) to specific host types, and then initiate processes in a round-robin fashion on those hosts. This would not perform as well, but may be an acceptable fall back option. Initial experiments and demos will operate in this mode.
  6. What is the minimum acceptable level for dynamic response of the system?
  7. How can access to resources such as disk space and I/O channel capacity be coordinated for applications that need a large fraction of the available capacity?
- **Task F2 - Distributed Scheduling:**
  1. What is the appropriate scheduling policy for intelligent agents that manipulate text? Can they be treated as small-sized tasks that can be executed on any platform?

2. What degree of assured service availability (fault tolerance) is required for the scheduling system? Are there upper limits to the execution time that should be met by the scheduler?

3. Will resource utilization be quantifiable for all systems of interest? In particular, it may be necessary to measure whether there is enough disk space to run an application or save results.

- **Task F3 - Integration of queuing system into Legion scheduling system:**
    1. Will it be possible to requeue jobs if a particular compute platform becomes unavailable? This is important to avoid excessive wait times, but runs the risk of an agent being executed twice

    2. What degree of assured service availability (fault tolerance) is required for the scheduling system when interacting with batch systems? Are there upper limits to the execution time that should be met by the scheduler?

    3. Will the queue waiting time predictions be sufficiently robust that reasonable estimates can be produced?

    4. Will resource utilization be quantifiable for all systems of interest? In particular, it may be necessary to measure whether there is enough memory or enough nodes available on a parallel computer to run the application.

- **Task F4 - Workflow performance testing and analysis:**
    1. Can the correct workflow parameters be defined that adequately specify all the characteristics of the work load?

    2. Can the correct performance metrics be specified for understanding the workload performance?

    3. How can security performance be quantified? One possibility is to quantify the time needed to detect an intrusion.

    4. How can fault tolerance performance be quantified? One possibility is to measure how the system responds to decreasing resource availability for fixed load.

    5. How can response time be measured? This may require instrumenting the intelligent agents to record state information about the start and stop execution times.

    6. How can the maximum sustainable workload be quantified as a function of the available resources? Such characterization usually rely on increasing the workload until the response time exponentiates.

    7. How can the effects of geographic separation be quantified? Will the response of the system be noticeable different for agents distributed from the east to west coasts versus from the east coast to NCSA?

G. Develop intelligent software agents

Task G1: Familiarization with the DOCT Environment

- **Task G1-1 - Determine Agent Framework Requirements in the DOCT Environment:**

    The agent framework will affect all issues that involve software agents, as the framework will define how the agents will interact with the DOCT environment. The following issues will most likely require agent support, and therefore will be affected by the framework:

    - Text search capabilities
    - Image search capabilities

- Search results display
- Rendering of documents
- Multimedia documents
- Intelligent agents
- Name management
- Task scheduling
- Fault tolerance
- API capabilities
- Distributed databases
- Confidentiality & intrusion
- Assured service availability
- Data integrity and auditability
- Access Management
- Version Management
- Recoverability
- Data Integrity

- **Task G1-2 - Prototype sample software agent framework and agent in DOCT environment using Legion, Open Text, DMS, and database archive:**

  This addresses a new issue under the document management framework issues:
  - Interoperability of DOCT components

Task G2 : The Software Agent Framework

- **Task G2-1 Requirements Analysis: Determine USPTO agent applications and develop an agent framework for the DOCT Environment**

  These issues are addressed from the perspective of agent applicability in performing processes for each issue:
  - Legacy document
  - Text search
  - Image search
  - Search result display
  - Version management
  - Confidentiality and intrusion
  - Originator authentication
  - Non-repudiation
  - Assured service (fault tolerance)
  - Data integrity and auditability
  - Secure date/time stamping
  - Access management
  - Version management
  - Persistence
  - Recoverability
  - Auditability
  - Data integrity

In addition the issue of intelligent agents for workflow support will be directly addressed as part of this task.

- **Task G2-2 High Level Design: Study current tools and agent frameworks available; design the DOCT agent framework and initial agents**

   This task will address the following issues, as they relate to software agents:

   - Text search
   - Image search
   - Search result display
   - Version management
   - Rendering of documents
   - Multimedia documents
   - Intelligent agents for workflow support
   - Confidentiality and intrusion
   - Originator authentication
   - Assured Service (fault tolerance)
   - Data integrity and auditability
   - Secure date/time stamping
   - Access management
   - Auditability
   - Data integrity

- **Task G2-3 - Detailed Design/Implementation: Develop agent framework and initial agents based on build 2 design**

   This task will address all issues identified in task G2-2.

Task G3 Software Agent Development

- **Task G3-1 Requirements Analysis/High Level Design: Update requirements and design based on build 2 results, add additional agents:**

   This task will address the issued identified in G2-1 and G2-2.

- **Task G3-2 - Detailed Design/Implementation: Update agent framework and develop new agents:**

   See issues from task G2-3.

Task G4 Advanced Agent Applications

- **Task G4-1 -Requirements Analysis: Study advanced agent applications and determine impact on current agent framework:**

   See task G2-1 for issues list.

- **Task G4-2 - High Level Design: Update design based on build 3 results and advanced requirements, design advanced agents:**

   See task G2-2 for a list of issues addressed.

Task G5 : Detailed Design/Implementation & Final Demonstration: Update agent framework, prototype advanced agents, demonstrate capabilities

- **Task G5-1 - Build advanced agents for the final demonstration:**

   See task G2-3.

H. Implement planning & management tools and procedures

- **Task H1 - Develop and Maintain a Detailed Research and Development Plan and Schedule:**

   There is one primary management issue relevant to the RDPS - the timely integration of seven separate cutting-edge, high performance computing and

communication technology R&D streams. These technology streams require the design, development, and integration of the following models:
- An overall Distributed Object Computational Testbed Architecture
- A DOCT data model
- A DOCT persistent object model
- A DOCT archive/storage model
- A DOCT assured service model
- A DOCT resource sharing/load sharing model
- A set of DOCT intelligent agent models

- **Task H2 - Develop and maintain a "Concept of Operations":**

  The key issue is how to show the relevance of the DOCT project to the mission and needs of other federal agencies. This is important because additional federal agency participation (and funding) will provide the option of expanding the scope and/or accelerating the R&D schedule for the project.

- **Task H3 - Provide Overall Management & Coordination:**

  Coordination and management of this project is complicated by its complexity, short duration, and remote location of the participants. Therefore we are adopting a management approach which emphasizes communication and coordination, i.e. detailed planning and scheduling (the RDPS document), a centralized document handling system to control the creation and distribution of both management and technical reports, a project management software package to help monitor and track both schedule and resource utilization, a standardized VTC system to facilitate remote communications between participants and sponsors, biweekly status meetings, and both monthly and quarterly progress reports.

- **Task H4 - Provide Scheduled Demonstrations:**

  The demonstration schedule is a target to present demonstration tasks as they become functional within the DOCT testbed. Because this is an R&D project, we anticipate both technical difficulties and modification of priorities which will cause individual task demonstrations to be modified, delayed, or, in some cases, canceled. In spite of the difficult nature of this project in terms of complexity, short duration, and remote location of the participants, we do hope to meet the schedule presented within this RDPS for all task deliverables, including the demonstrations.